CHALMERS

Model Predictive Control of a Quanser 3DOF Helicopter

Double Master's Thesis

Markus Ruf

Supervisors:
Oskar Wikstrom, Chalmers University of Technology
Florian Bayer, University of Stuttgart

Examiners:
Ass. Prof. Balázs Kulcsár, Chalmers University of Technology
Prof. Dr.-Ing. Frank Allgöwer, University of Stuttgart

Department of Signal and System
CHALMERS UNIVERSITY OF TECHNOLOGY

Institute for Systems Theory and Automatic Control
UNIVERSITY OF STUTTGART

Munich, Germany 2014

Model Predictive Control of a Quanser 3DOF Helicopter

This thesis was conducted within the Double Master's Programme in System, Control and Mechatronics at Chalmers University of Technology, Gothenburg, Sweden and Cybernetic Engineering at University of Stuttgart, Germany. The responsible departments are the Department of Signal and System at Chalmers University of Technology and the Institute for System Theory and Automatic Control at University of Stuttgart.

Master's Thesis 2014:02

Department of Signal and System
Chalmers University of Technology
SE-41296 Göteborg
Sweden
Tel. +46-(0)31 772 1000

Institute for Systems Theory and Automatic Control
University of Stuttgart
DE-70569 Stuttgart
Germany
Tel. +49-(0)711 685 67734

Abstract

Model Predictive Control has been widely used in industrial practise, but has been restricted to applications with relatively slow dynamics in the past due to limited computational power. Its capability of handling constraints easily makes it very attractive for a wider range of application though.

The purpose of this thesis is to investigate Model Predictive Control for a 3DOF Laboratory Helicopter, which offers fast dynamics and nonlinearities which may be unknown. A nonlinear mechanical model of the helicopter is developed and model parameters are identified. After linearisation the model is split into three parts, one for each axis. Individual Model Predictive Controllers are implemented in a cascation structure to decrease computational complexity. Input disturbance estimators are used to achieve zero-offset control. Nonlinear effects are included in the MPC as measured disturbances. A terminal weight and terminal set are implemented in order to prove stability. Experimental results are employed to investigate the controller's performance and properties.

The results show that an excellent controller performance can be achieved with MPC, mainly because the constraints can be enforced easily. It is not possible though to formally show stability within a reasonably large feasible set as control and prediction horizon are very restricted due to computational limitations. This shows that computational limitations are still the main challenge for proofing stability of Model Predictive Controllers in practise.

Contents

List of used symbols and shortcuts

MPC	Model Predictive Control
LQR	Linear Quadratic Regulator
3DOF	3 Degrees of Freedom
MPT	Multi-Parametric Toolbox
$\mathcal{X}$	State Constraint Set
$\mathcal{U}$	Input Constraint Set
Q	State Weight
R	Input Weight
H_p	Prediction Horizon
H_c	Control Horizon

List of Figures

List of Tables

1 Introduction

Model Predictive Control is an optimal control method which originates in the process industry. The core idea of Model Predictive Control is to repeatedly solve a finite horizon optimal control problem at every time step, but only apply the very first input signal supplied by the optimisation algorithm. This process is then repeated for every proceeding time step. The main advantage of Model Predictive Control is that no explicit control law is needed as only a finite optimal control problem is solved at every time step using a numerical optimisation algorithm. Consequently, system constraints can be handled very easily, which is often impossible for infinite optimal control. In practise this can be very beneficial, as many plants are most effectively operated near their physical constraints. Unfortunately, the main advantage of Model Predictive Control is also its biggest flaw: As an optimal control problem has to be solved at every single time step, Model Predictive Control is extremely computationally demanding. This is especially the case for systems with fast dynamics and thus small sampling intervals, which leaves very few time for the optimisation algorithm to solve the problem. This is the main reason why Model Predictive Control had been restricted to the process industry in the past where the dynamics are very slow, often with sampling intervals of several minutes. But with much more computational power nowadays, Model Predictive Control becomes available to much faster systems. Nevertheless, computational demand remains the main issues in Model Predictive Control. An additional challenge in Model Predictive Control is to systematically show stability. When Model Predictive Control started in the process industry, no formal stability theory was available. Over the last decades a lot of research was done on the topic which resulted in a mainly Lyapunov based stability theory for Model Predictive Control. Nevertheless, this stability theory is hardly used in industrial practise to this day. One of the reasons is that this stability theory often has additional computational demands.

This thesis investigates the implementation of a Model Predictive Controller on a system with fast dynamics and considerable nonlinear effects. The main focus lays on how to deal with the computational demand and the performance and practicability of the Model Predictive Controller compared to traditional methods. Additionally, the stability theory is implemented for this application and conclusions are drawn on its feasibility in practise. A Quanser helicopter with 3 degrees of freedom will be used as a benchmark system. It consists of two rotors mounted on an arm which can be elevated up and down and rotated around a vertical axis. The helicopter offers relatively fast dynamics and due

to its multibody character considerable nonlinear effects. As it is used by universities all over the world in in laboratories and to benchmark different kind of controllers, it is ideal to compare Model Predictive Control to other methods.

2 Experimental Setup

The 3 degrees of freedom (3DOF) helicopter is designed and manufactured by the Canadian company Quanser. It is depicted in Figure 2.1. The helicopter consists of a base on which a turn-able axis is mounted vertically (A in Figure 2.1). An arm is attached on the top of that vertical axis (B). This arm can be moved up and down. The actual helicopter is attached to the front of this arm (C). It can be pitched up and down. Addtionally, a counterweight is attached to the back of the arm (D).

There are 3 incremental angle sensors, one for each axis ((s) in Figure 2.1). Both rotors can be steered individually.

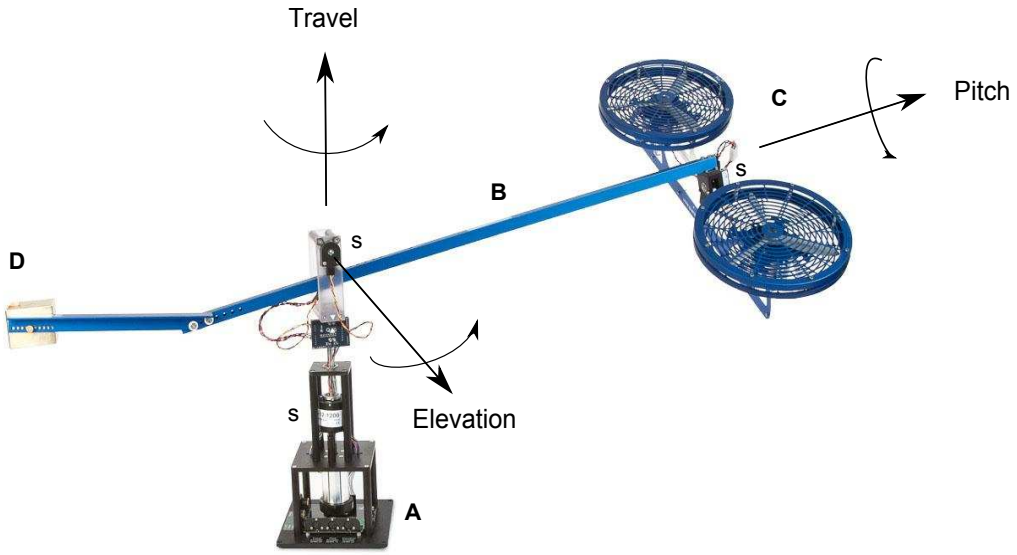

Figure 2.1: Quanser 3DOF Helicopter (Picture Source: [exp]).

The helicopter can be controlled using Quanser's Rapid Prototyping Software and Hardware, which is called QUARC. It provides an interface to Matlab-Simulink. The Simulink models need to be compiled into C-Code to work on QUARC.

3 Model Identification

In this chapter, a coordinate system and conventions for the Quanser 3DOF Helicopter are defined. Then a nonlinear model structure is derived.

In the second part of the chapter, model parameters are identified and analysed using several experiments conducted on the real plant.

In the third part, the model is verified by comparing open-loop simulation and experiments on the real plant.

3.1 Modeling

3.1.1 Conventions and Coordinate Systems

Distances

All geometric distances will be labeled with d and are measured in meters. The index will specify to which axis the distance belongs, for example $d_{\theta h}$ belongs to the θ axis. The other index h indicates that $d_{\theta h}$ is referring to a height.

Angles

All angles will be labeled by Greek letters.

Identified Parameters

All parameters which are identified experimentally are labeled with a k. The index specifies the purpose of the parameter, for example k_{thrust}.

Momentums

All momentums are referred to with a capital M. The index will indicate the cause of the momentum in the first letter. There are 3 types of causes:

1. M_g: Momentums caused by the gravitational force
2. M_m: Momentums caused by the motors
3. M_r: Momentums caused by relative forces in the multibody system.

3.1.1.1 Coordinate Systems

Pitch Axis

- Positive θ if the front rotor goes down (the front motor is labeled on the helicopter)
- Zero if parallel to the ground
- Negative θ if the front rotor goes up

Elevation Axis

- Negative ϕ if the helicopter is below the joint.
- Zero ϕ if bottom of the helicopter is parallel to the joint of the elevation axis.
- Positive ϕ if helicopter is above the joint of the elevation axis.
- The sensors are initialised on the ground, so an elevation offset has to be programmed in the Simulink model. The offset is $\phi_0 = -0.7164 rad$.

Travel axis

- Positive ψ if rotating in the right hand sense (as depicted in Figure 2.1)
- Zero ψ at starting point
- Negative ψ if rotating in the left hand sense

3.1.2 Rotor Thrust

The aerodynamic lift produced by the rotors depends quadratically on their rotational rates ω_f and ω_b respectively. This is due to the fact that the aerodynamic lift of an object depends on its squared speed and two constant factors:

$$F_{lift} = \frac{1}{2} c_l A v^2,$$

where c_l is an aerodynamic coefficient, A the effective surface area and v the speed of the object. For further details, please consider [Ste03]. The rotational rate of the rotors depends on the impressed voltage and the motor dynamics. As the motor dynamics are very fast, they are neglected. As a result, the lift forces can be described by

$$\begin{aligned} u_f &= k^f_{thrust} U_f^2 \\ u_b &= k^b_{thrust} U_b^2, \end{aligned} \tag{3.1}$$

where U_f is the voltage impressed on the front motor and U_b on the back motor respectively. The factors k^f_{thrust} and k^b_{thrust} have to be identified experimentally.

It is convenient to define the sum force and the difference force of the motors as inputs to the system:

$$\mathbf{u} = \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} = \begin{bmatrix} u_f + u_b \\ u_f - u_b \end{bmatrix}. \tag{3.2}$$

This yields

$$u_f = \frac{1}{2}(u_{sum} + u_{diff})$$
$$u_b = \frac{1}{2}(u_{sum} - u_{diff}).$$

3.1.3 Pitch

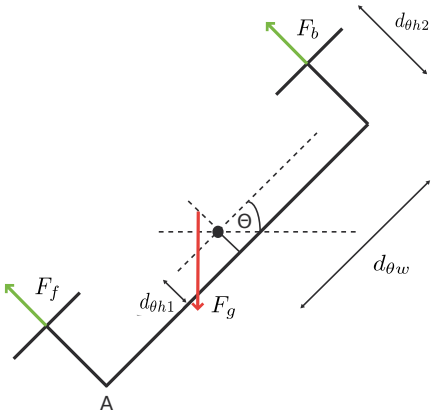

Figure 3.1: Outline of the Pitch Axis.

On the pitch axis, there are three main forces: the thrust of the front motor u_f, the thrust of the back motor u_b and the gravitational force F_g. There are additional relative forces such as the Coriolis force, since the helicopter is rotating around the travel and elevation axis.

Those forces result in momentums around the pitch axis. The momentum around the joint produced by the motors can be described by

$$M_{mdiff} = d_{\theta w}u_b - d_{\theta w}u_f = d_{\theta w}(\underbrace{u_b - u_f}_{u_{diff}}).$$

Because the joint is not attached to the center of mass of the helicopter, the gravitational force also results in a momentum around the joint. It is difficult to determine the center

of mass. The momentum could be determined experimentally though. To do so, the weight of the helicopter at point A (See Figure 3.1) for different pitch angles will be measured. The momentum $M_{g\theta}$ around the joint can then be calculated as follows:

$$M_{g\theta} = \underbrace{\sqrt{d_{\theta h1}^2 + d_{\theta w}^2}}_{d_{\theta h}} sin(\theta) F_{gmeas}(\theta),$$

where $F_{gmeas}(\theta)$ is the measured force at point A.
It also turned out that the weight on the pitch axis is not distributed symmetrically. The helicopter has the tendency to tilt to the side of the front motor. This means that the helicopter's front motor is slightly heavier or that it is mounted slightly further from the joint.

Additionally, the Coriolis Effect has to be taken into account. Because of this effect, the pitch angle tends to go to zero, when the helicopter is rotating around the travel axis. It depends on the travel speed and the pitch angle.

[Wol11] also modeled friction in the pitch joint, but it is very small. It will be neglected here.

The inertia J_p around the pitch axis has been determined by [Eit11].

Using the principle of conservation of momentum, those considerations result in the following model for the pitch axis:

$$\begin{aligned} J_p\ddot{\theta} &= M_{mdiff}(u_{diff}) + M_{g\theta}(\theta) + M_{rcor}(\theta, \dot{\psi}) \\ &= d_{\theta w} u_{diff} + d_{\theta h} sin(\theta_0) F_{meas} + M_{rcor}. \end{aligned}$$

3.1.4 Elevation

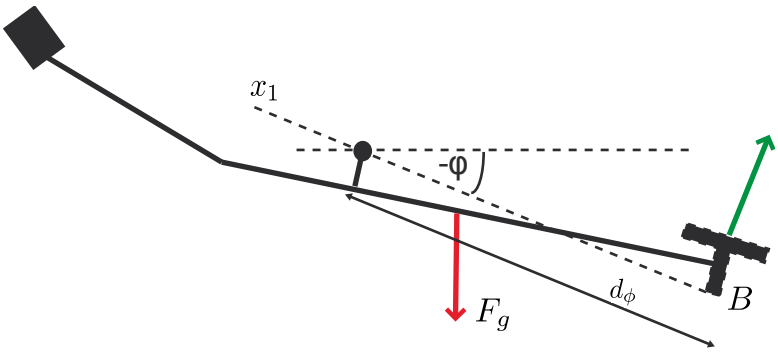

Figure 3.2: Outline of the Elevation Axis.

On the elevation axis, there are two forces involved: the gravitational force F_g and the part of the thrust force of the motors F_m^v vertical to the x_1 axis.

The thrust force F_m^v causes a momentum around the joint, which can be described by

$$M_{mthrust\phi} = d_\phi cos(\phi) \underbrace{(F_f + F_b)}_{u_{sum}}.$$

The gravitational force F_g causes a gravitational momentum $M_{g\phi}$. [Wol11] tried to model this momentum using geometry. Nevertheless, it is more practical and precise to obtain this static momentum by measuring the force at point B (see Figure 3.2) for different angles ϕ using a weighting scale. With this measurements, the gravitational momentum with respect to the joint can be obtained by

$$M_{g\phi} = d_\phi cos(\phi) F_{mg}(\phi).$$

[Wol11] also modeled friction on this axis, but it is very small and will not be modeled here.

Because of the rotation around the travel axis the relative forces are relevant on the elevation axis. This rotation creates a centrifugal force which drives the helicopter upwards. [Wol11] used geometry to calculate this centrifugal force. To do so, he had to make some approximations. As the centrifugal force only depends on the rotation around the travel axis and a constant factor, it seems more feasible to determine it experimentally though. In the experiment, the helicopter is rotated around the travel axis. After some time, it will set to a certain elevation angle ϕ. Then, the part of the centrifugal force perpendicular to the x_1 axis is equal to the gravitational force for this elevation. The centrifugal force can be described by

$$M_{rcen} = k_{cen} \dot{\psi}^2,$$

where k_{cen} is a scalar constant and $\dot{\psi}$ the rotational speed around the travel axis. The details of this experiment are explained in the corresponding chapter on identification (3.2.4).

Another important momentum is caused by the rotors themselves. Because both rotors are mounted in the same direction, air drag causes a torque around the rotational axis of the rotors. Air drag can be described by

$$F_{drag} = \frac{1}{2} c_d \rho A v^2,$$

where c_d is the aerodynamic drag coefficient, ρ the density of the air and A the effective surface area of the rotors.

[Wol11] has shown that the drag momentum depends quadratically on the rotor's thrust. Thus

$$\begin{aligned} M_{mdrag} = k_{m\,drag}(u_f^2 + u_b^2) &= \frac{1}{4} k_{m\,drag}((u_{sum} + u_{diff})^2 + (u_{sum} - u_{diff})^2) \\ &= \frac{1}{2} k_{m\,drag}(u_{sum}^2 + u_{diff}^2) \end{aligned}$$

where k_{mdrag} is a constant factor to be identified.

So the effective torque on the elevation axis is

$$M^e_{m\,torque} = \frac{1}{2} k_{mdrag} sin(\theta)(u_{sum}^2 + u_{diff}^2).$$

Using the principle of conservation of momentum, those considerations result in the following model for the elevation axis:

$$J_e \ddot{\phi} = M_{m\,thrust\phi}(u_{sum}) + M_{g\phi}(\phi) + M_{rcen}(\dot{\psi}) + M^e_{m\,torque}(u_{sum}, u_{diff}) \tag{3.3}$$

$$= d_\phi cos(\phi) u_{sum} + d_\phi F_{mg}(\phi) + k_{cen}\dot{\psi}^2 + \frac{1}{2} sin(\theta) k_{m\,drag}(u_{sum}^2 + u_{diff}^2). \tag{3.4}$$

3.1.5 Travel

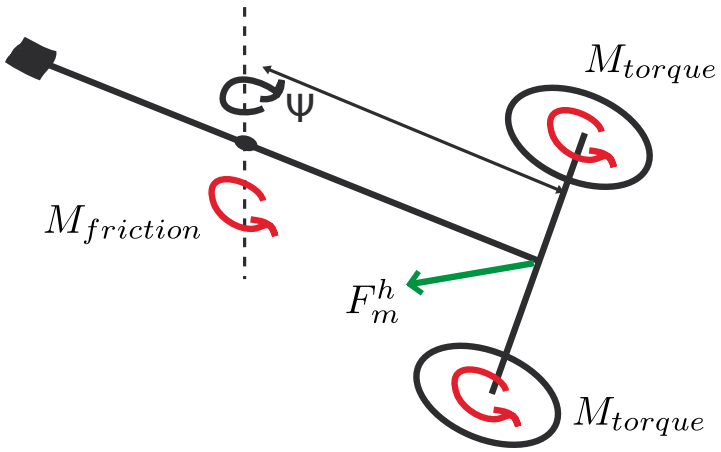

Figure 3.3: Outline of the Travel Axis.

Around the travel axis, there are two significant forces. The first one is the thrust force of the engines perpendicular to the travel axis. This force can be approximated by

$$F_m^h = sin(\theta) u_{sum} TerminalConstraint$$

(ignoring the fact, that the joint is not directly attached to the arm, but slightly above). This yields the momentum

$$\begin{aligned} M_{m\,thrust\psi} &= d_\psi F_m^h \\ &= d_\psi sin(\phi) u_{sum} \end{aligned}$$

As for the elevation axis, the rotor's torque has an impact on the behaviour on the travel axis. It can be described by

$$M_{m\,torque}^t = \frac{1}{2} cos(\theta) k_{mdrag} (u_{sum}^2 + u_{diff}^2).$$

Additionally, the friction on the travel axis is notable. Therefore, a linear friction term will be assumed:

$$M_{friction} = -k_{friction} \dot{\psi},$$

where $k_{friction}$ is a constant factor to be determined.

The parameter $k_{friction}$ can be obtained with the same experiment used to identify the centrifugal force. There, the helicopter is rotating around the travel axis. Because of the

friction, the rotational speed decreases. This can be used to determine the friction: If the motors are offline, it can be said that

$$\begin{aligned} J_t\ddot{\psi} &= M_{friction} \\ &= -k_{friction}\dot{\psi}. \end{aligned}$$

Solving this differential equation yields

$$\dot{\psi}(t) = \dot{\psi}(0)e^{-\frac{k_{friction}}{J_t}t}.$$

This function can then be fitted into the measurements of the experiment to obtain d.

All together, the angular momentum around the travel axis can then be described as

$$J_t\ddot{\theta} = M_{m\,thrust\psi}(\theta,\phi) + M^t_{m\,torque}(\theta, u_{sum}) + M_{friction}(\dot{\psi}) \tag{3.5}$$

$$= d_\psi sin(\theta)u_{sum} + \frac{1}{2}cos(\theta)k_{m\,drag}(u^2_{sum} + u^2_{diff}) - k_{friction}\dot{\psi}. \tag{3.6}$$

3.1.6 Summary

With the models for each individual axis derived above, a representation of the helicopter dynamics can be summed up by

$$\begin{aligned} J_p\ddot{\theta} &= M_{mdiff}(u_{diff}) + M_{g\theta}(\theta) + M_{rcor}(\theta,\dot{\psi}) \\ J_e\ddot{\phi} &= M_{m\,thrust\phi}(u_{sum}) + M_{g\phi}(\phi) + M_{rcen}(\dot{\psi}) + M^e_{m\,torque}(u_{sum}, u_{diff}) \\ J_t\ddot{\theta} &= M_{m\,thrust\psi}(\theta,\phi) + M^t_{m\,torque}(\theta, u_{sum}) + M_{friction}(\dot{\psi}) \end{aligned}$$

or

$$\begin{aligned} J_p\ddot{\theta} &= d_{\theta w}u_{diff} + d_{\theta h}sin(\theta_0)F_{meas} + M_{rcor} \\ J_e\ddot{\phi} &= d_\phi cos(\phi)u_{sum} + d_\phi F_{mg}(\phi) + k_{cen}\dot{\psi}^2 + \frac{1}{2}sin(\theta)k_{m\,drag}(u^2_{sum} + u^2_{diff}) \\ J_t\ddot{\psi} &= d_\psi sin(\theta)u_{sum} + \frac{1}{2}cos(\theta)k_{m\,drag}(u^2_{sum} + u^2_{diff}) - k_{friction}\dot{\psi} \end{aligned}$$

respectively.

This provides the following State Space model

$$
\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{1}{J_p}(d_{\theta w}u_{diff} + d_{\theta h}sin(\theta)F_{meas} + M_{rcor}) \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= \frac{1}{J_e}(d_{\phi}cos(x_1)u_{sum} + d_{\phi}F_{mg}(x_3) + k_{cen}x_6^2 + \frac{1}{2}sin(x_3)k_{mdrag}(u_{sum}^2 + u_{diff}^2)) \\
\dot{x}_5 &= x_6 \\
\dot{x}_6 &= \frac{1}{J_t}(d_{\psi}sin(x_1)u_{sum} + \frac{1}{2}cos(x_1)k_{mdrag}(u_{sum}^2 + u_{diff}^2) - k_{friction}x_6)
\end{aligned}
\tag{3.7}
$$

where $x_1 = \theta$, $x_2 = \dot{\theta}$, $x_3 = \phi$, $x_4 = \dot{\phi}$, $x_5 = \psi$, $x_6 = \dot{\psi}$. Following definition (3.2), u_{sum} and u_{diff} are inputs describing the sum and differential of the thrust force of the rotors.

3.2 Parameter Identification

3.2.1 Thrust Force

For the thrust force, the parameters k^b_{thrust} and k^f_{thrust} in equation (3.1) have to be identified.

To do so, the motor's thrust is measured using a weighting scale for increasing impressed voltages. This seems like a simple task, there are some problems to deal with though:

1. The individual motors do not produce the same thrust for equal impressed voltages. Consequently, k^b_{thrust} and k^f_{thrust} can not be identified at the same time by measuring the sum force of the motors. As a results, the motors had to be dismounted from the helicopter to be able the measure their thrust independently (See Figure 3.4)
2. If rotors are operated very close to the ground, the so called ground effect causes an increase in the rotor thrust (further details see [Ste03]). This is normally not problematic for the laboratory helicopter, as it will be operated high enough above the ground. For the identification though, the motors have to be mounted to the weighting scale (See Figure 3.4). So they are close to the ground and the ground effect is disturbing the measurements consequently.

The experiment results in the relation depicted in Figure 3.5. One can clearly see that

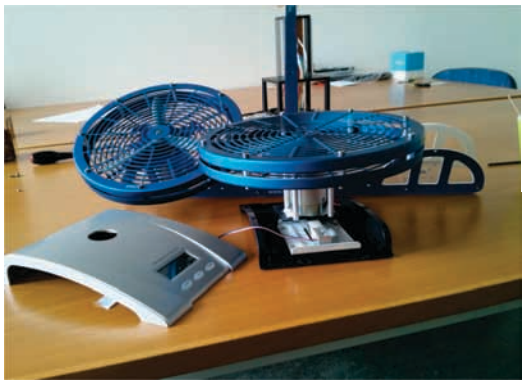

Figure 3.4: Thrust Identification: Detached Rotor and Weighting Scale.

the front motor produces more thrust for the same impressed voltage than the back motor. Because of the proximity of the rotors to the ground, the relation in Figure 3.5 has a certain offset from the real values due to the ground effect. To deal with this, it is assumed that the ground effect acts proportionally to the provided thrust, thus

$$F_{G.effect} = k_g F_{thrust}.$$

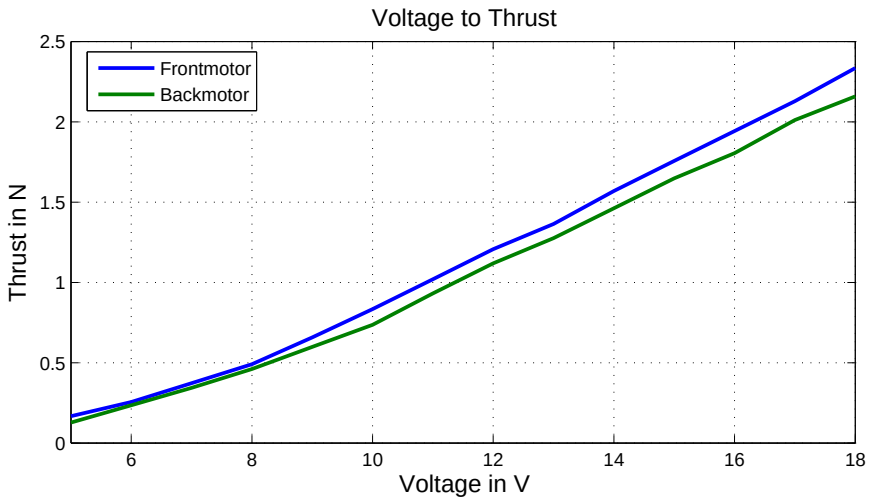

Figure 3.5: Measured Thrust Levels for Different Voltages.

By comparing the relation in Figure 3.5 to the gravitational momentum on the elevation axis in hovering experiments, it turned out that the ground effect is rather significant with $k_g = 1.42$.

Another problem was that the measured difference of the motors had some small errors. To compensate, the thrust of the back motor was multiplied with a correction factor of $k_c = 0.98$ (see Figure 3.6).

Because relation (3.1) is static, it is not directly part of the model used in the controller. Instead is has been implemented as a Look-Up-Table in the Simulink interface used to control the helicopter (see Figure 3.6). Therefore k^b_{thrust} and k^f_{thrust} were not determined explicitly.

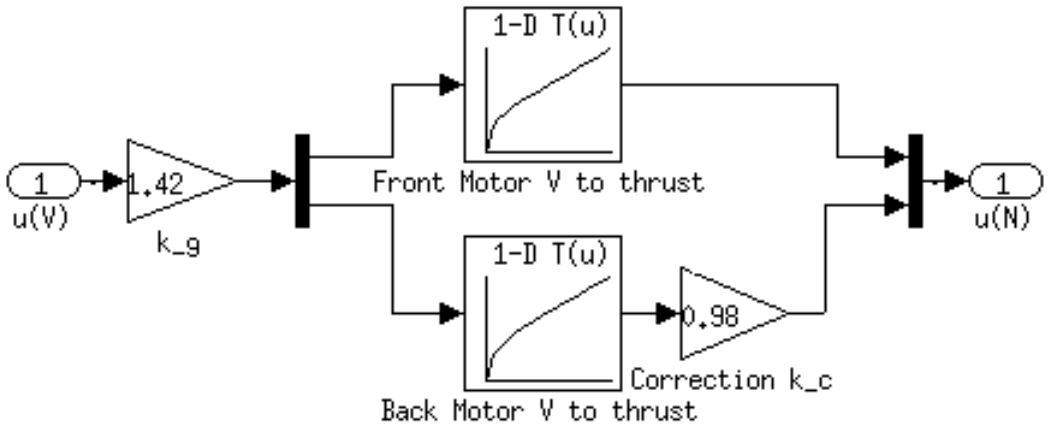

Figure 3.6: Simulink Implementation Thrust Force.

3.2.2 Inertias

The intertias used in this thesis have been experimentally identified by [Doh11], see Table 3.1. In these experiments, disks had been added to each axis. Specific weights had then been attached to those disks. Using step experiments, the intertias have then been identified. For further information see [Doh11], page 32.

Table 3.1: Intertias Identified by [Doh11].

Parameter	Value
J_t	$1.1785 kgm^2$
J_p	$0.0398 kgm^2$
J_e	$1.1555 kgm^2$

3.2.3 Pitch

Gravitational Force $M_{g\theta}$

As mentioned before, there are two effects causing the gravitational force on the pitch axis:

- Center of mass is not in the joint
- Front motor is heavier than the back motor

Unfortunately, as the gravitational force on the pitch axis is rather small, it was not possible to get accurate and consistent measurements with the available weighting scale. Therefore, the gravitational force will be neglected. Thus the gravitational momentum is assumed to be

$$M_{g\theta} = 0Nm.$$

Coriolis Force

Even though the Coriolis does have some impact, its identification was very difficult as it could not be investigated independent from other influences such as the centrifugal force and wind resistance. As it was not possible to gain an exact model of those forces either, the Coriolis force could not be isolated in experiments. Therefore it is assumed to be zero. The controller will later deal with this unknown force by using disturbance estimation methods.

Resulting Model on Pitch Axis

With the gravitational and the Coriolis force neglected, the model around the pitch axis simplifies to:

$$\dot{x}_1 = x_2 \tag{3.8}$$

$$\dot{x}_2 = \frac{1}{J_p} d_{\theta w} u_{diff} \tag{3.9}$$

3.2.4 Elevation

Gravitational Momentum $M_{g\phi}$

In order to determine the gravitational momentum $M_{g\phi}$, the gravitational force $F_{mg}(\phi)$ at point B (see Figure 3.1) has been measured for different elevation angles ϕ using a weighting scale. Those measurements can be interpolated by a linear function (see Figure 3.7) .

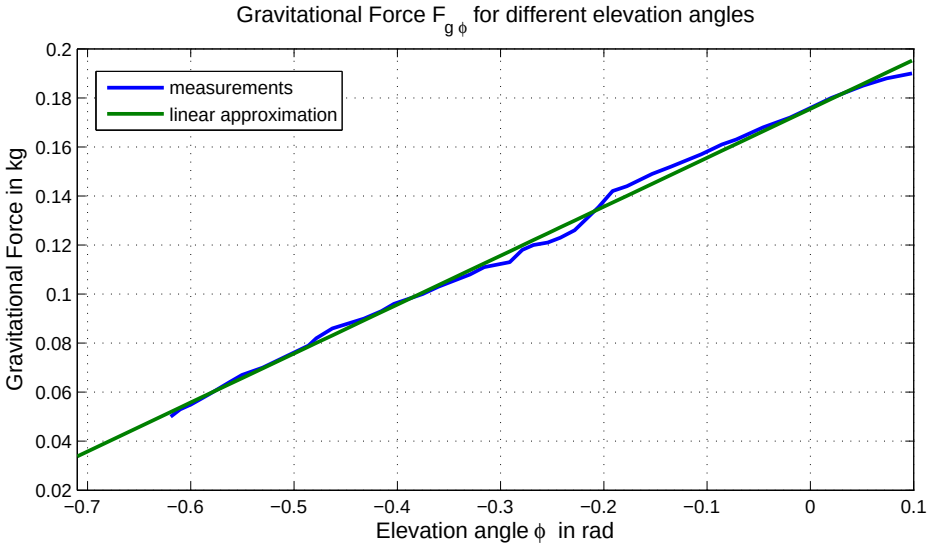

Figure 3.7: Measured Gravitational Force $M_{g\phi}$ for Different Elevation Angles.

Linear approximation:

$$F_{mg}(\phi) = (1.996(\phi + 0.71) + 0.3383)N$$

Consequently the gravitational momentum $M_{g\phi}$ can be described by

$$\begin{aligned} M_{g\phi}(\phi) &= d_\phi cos((\phi + 0.71))F_{mg}(\phi) \\ &= d_\phi cos((\phi + 0.71))(1.996(\phi + 0.71) + 0.3383)N \\ &= d_\phi cos((\phi + 0.71))(1.996\phi + 1.7555)N \end{aligned} \tag{3.10}$$

Centrifugal Force

In order to determine the centrifugal force, the following experiment was conducted: The helicopter is rotated around the travel axis. When rotating, the centrifugal force counteracts the gravitational momentum on the elevation axis, pushing the helicopter upwards. As a result, gravitational momentum and centrifugal momentum compensate each other and the helicopter reaches a certain equilibrium elevation angle for a certain travel speed. The results can be seen in Figure 3.9. This relation can be used to derive an estimation of the centrifugal force.

It is important to note that the centrifugal force is not acting directly on the elevation axis. It is acting perpendicular to the travel axis, pushing the helicopter outwards. If the arm is not perpendicular to the travel axis, a force F_{up} will then act upwards (see Figure 3.8). It can be described by

$$F_{up}(\phi, \dot{\psi}) = tan(\phi) F_{cen}(\dot{\psi}). \tag{3.11}$$

Here, F_{up} is compensating the gravitational momentum in the experiment. One might expect to be able to calculate F_{cen} using (3.11). That is not true though, since F_{up} depends on the elevation angle ϕ as well as the travel speed $\dot{\psi}$. The experiment only covers F_{up} for specific $\dot{\psi}(\phi)$. So $F_{up}(\phi, \dot{\psi})$ can not be calculated, but only its projection on the subspace spanned by $\dot{\psi}(\phi)$. Thus,

$$P_{\dot{\psi}(\phi)} F_{up}(\phi, \dot{\psi}) = F_{g\phi}(\phi)$$

where $P_{\phi(\psi)}$ is the projection on $span(\phi(\psi))$.

Nevertheless, $P_{\dot{\psi}(\phi)} F_{up}(\phi, \dot{\psi})$ still provides useful information, as it results in a lower bound for the elevation angle.

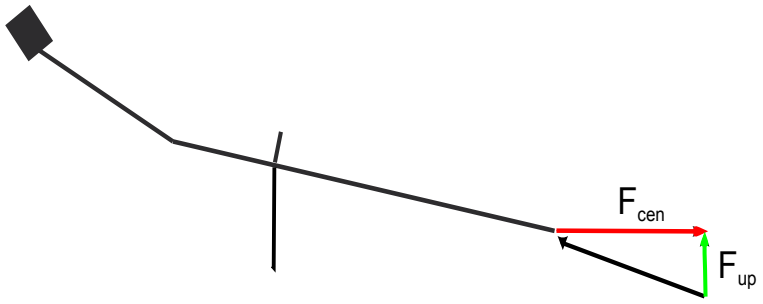

Figure 3.8: Outline of the Centrifugal Force.

The oscillation in Figure 3.9 occurs, because the helicopter goes up and down during the rotation. The reason is that the initial rotational speed of the helicopter is not exactly

given. Consequently, the uplifting centrifugal momentum is smaller or bigger than the corresponding gravitational momentum for the associated angle. So, if the gravitational momentum is larger, the helicopter falls down a bit until the gravitational momentum is small enough.

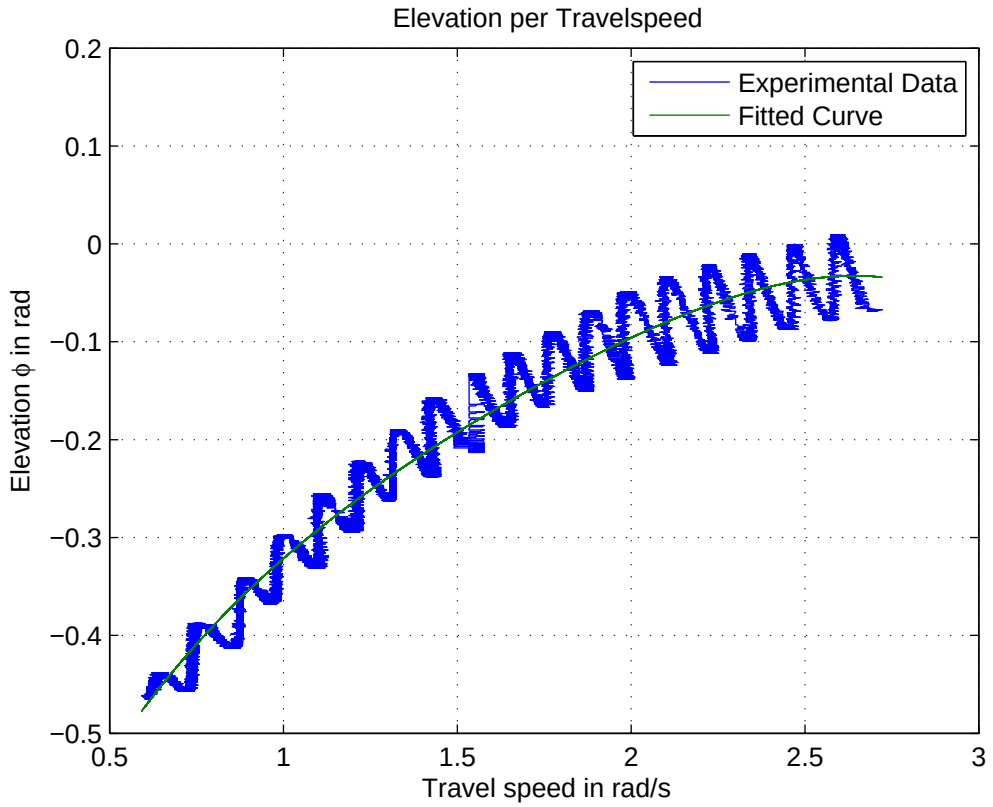

Figure 3.9: Measured Elevation Angle for Different Travel Speeds.

Then it goes up again, etc. The mean value of that curve represents the desired relation between the rotation and uplift tough. The relation between the elevation and travel speed can then be approximated by the following function (green curve in Figure 3.9):

$$\phi(\dot{\psi}) = -0.03061\dot{\psi}^4 + 0.2074\dot{\psi}^3 - 0.5771\dot{\psi}^2 + 0.9639\dot{\psi} - 0.8851 \tag{3.12}$$

The gravitational force for different elevation angles has been identified before (Equation (3.10)). This can be matched with the above curve:

$$P_{\dot{\psi}(\phi)} F_{up}(\phi, \dot{\psi}) = cos((\phi(\dot{\psi}) + 0.71))(1.996(\phi(\dot{\psi}) + 0.71) + 0.3383)N$$

and consequently

$$P_{\dot{\psi}(\phi)} M_{up}(\phi, \dot{\psi}) = d_\phi cos((\phi(\dot{\psi}) + 0.71))(1.996(\phi(\dot{\psi}) + 0.71) + 0.3383)N.$$

Torque

Rotating rotors create an aerodynamic lift force. But they are also exposed to aerodynamic drag which results in a torque opposite to the direction of the rotation of the rotors. On helicopters with two rotors, this is normally compensated by rotating the rotors in the opposite direction. For the Quanser helicopter though, they are mounted in the same direction. A considerable torque results from this.

To identify this torque, two approaches were taken:

1. Measuring the torque with a weighting scale
2. Closed-loop identification

Weighting scale:

The first approach taken to identify the torque is to measure the corresponding force on the arm on which the helicopter is mounted. To do so, the weighting scale was mounted vertically next to the arm. Consequently, the torque force can be measured for different thrust levels, as the helicopter is pushing against the weighting scale. This torque force is rather small though, which made it very difficult to measure with a simple weighting scale, which is not precise enough. So this approach was not used after all.

Closed-loop identification:

The torque caused by the rotors is perpendicular to the rotors themselves. This means that if the pitch angle is zero, this torque will result in a acceleration around the travel axis. This acceleration can then be compensated by pitching the helicopter. This has two effects: The torque is not parallel to the travel axis any more and the motor's thrust is now causing a momentum around the travel axis. Those two effects can compensate for the acceleration around the travel axis. This was done for a range of sum forces. The results are depicted in Figure 3.10.

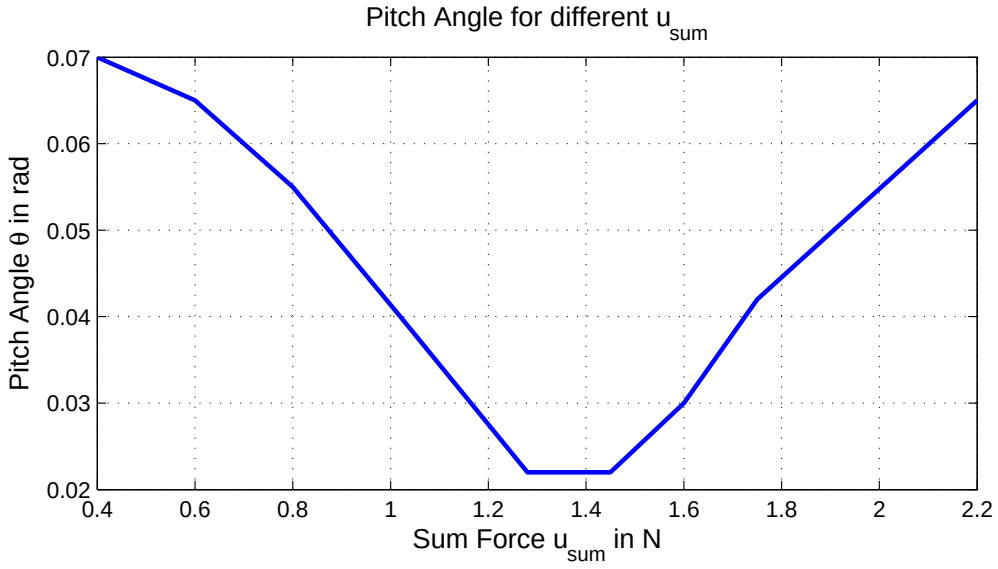

Figure 3.10: Pitch Angle Necessary to Prevent Acceleration on the Travel Axis.

It seems counter intuitive that the angle is not increasing over the whole range, because the drag force should increase with the sum force. But when the drag force is increasing with increasing u_{sum}, the thrust force is also increasing. So smaller pitch angles are needed to create a certain thrust momentum around the travel axis, namely

$$M_{back\theta} = \frac{d_\psi}{J_t} sin(x_1) u_{sum}.$$

Additionally, for $\theta \neq 0$ the drag momentum is not completely parallel to the travel axis, thus

$$M_{mdrag\psi} = cos(x_1) M_{mdrag}.$$

This results in

$$\begin{aligned} M_{mdrag\psi} &= M_{back\theta} \\ cos(x_1) M_{mdrag} &= \frac{d_\psi}{J_t} sin(x_1) u_{sum} \\ \Rightarrow M_{mdrag}(x_1, u_{sum}) &= \frac{d_\psi}{J_t} tan(x_1) u_{sum}. \end{aligned}$$

The resulting drag momentum is depicted in Figure 3.11.

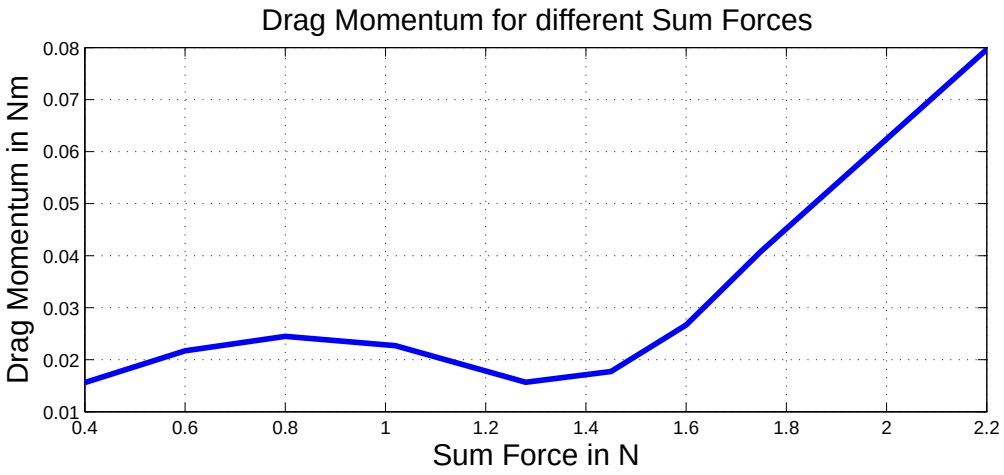

Figure 3.11: Identified Drag Momentum for Different Sum Forces.

It is notable that the drag force is not increasing over the whole range. Consequently, the torque can then be restricted to

$$\|M_{mdrag}\| \leq 0.08Nm$$

With neglected centrifugal force, the elevation axis model is simplified to

$$\dot{x}_3 = x_4 \tag{3.13}$$

$$\dot{x}_4 = \frac{1}{J_e}(\underbrace{d_\phi cos(x_1)u_{sum}}_{M_{mthrust\phi}} + \underbrace{d_\phi cos(\phi)(1.996\phi + 0.3383)N}_{M_{g\phi}} \tag{3.14}$$

$$+ \underbrace{\|M_{mdrag}\|sin(x_1)}_{M_{drag\phi}} \tag{3.15}$$

3.2.5 Travel Axis

Friction

The friction caused by the joint can be identified using the same experiments as for the centrifugal force. As mentioned in the previous chapter, the decline over time of the

elevation caused by the friction around the travel axis can be described by

$$\dot{\psi}(t) = \dot{\psi}(0)e^{-\frac{k_{friction}}{J_t}t}. \tag{3.16}$$

The fraction $-\frac{k_{friction}}{J_t}$ can be determined by fitting the curve (3.16) into the elevation over time plot of the experiment.

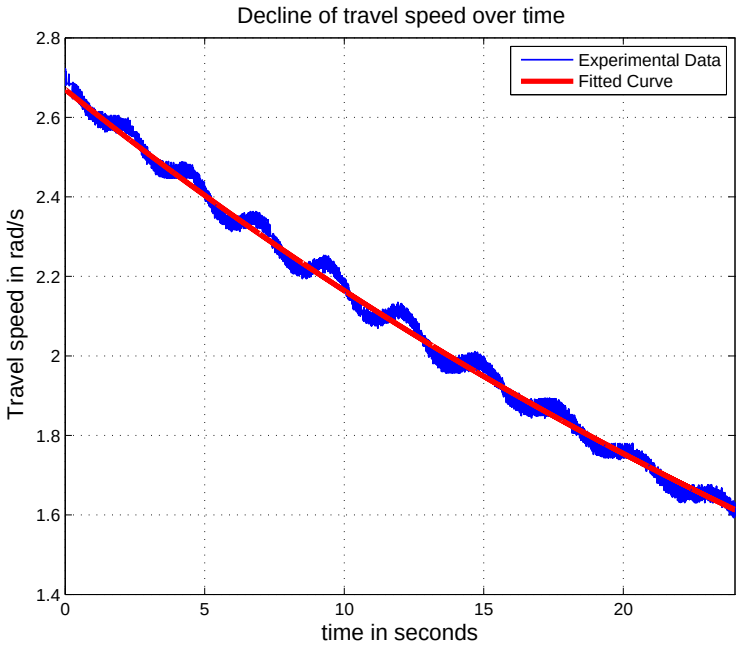

Figure 3.12: Decline of the Elevation Angle Over Time with Fitted Curve $\dot{\psi}(t) = 2.67e^{-0.021t}$.

This yields $-\frac{k_{friction}}{J_t} = -0.021\frac{1}{s}$. Using $J_t = 1.1785kgm^2$ as determined by [Doh11], it follows that

$$k_{friction} = 0.021\frac{1}{kgm^2}J_t = 0.0247\frac{rad}{s}. \tag{3.17}$$

During later experiment, it turned out that this is not the only significant friction: Wind resistance of the helicopter also has an important influence on the helicopter's behaviour. This wind resistance is dependent on the pitch angle. When the pitch angle is close to zero, the wind resistance is very small. For simplicity, wind resistance is then assumed to be zero. In contrast, the wind resistance is significantly larger for larger pitch angles.

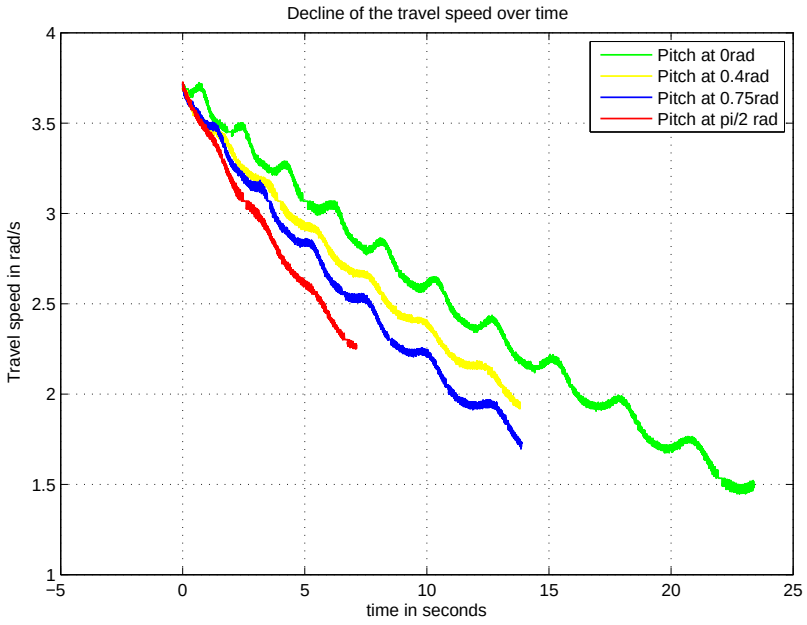

Figure 3.13: Decline of Travel Speed Over Time for Different Pitch Angles.

To get an idea about the magnitude of this wind resistance the above experiment was repeated for a couple of different pitch angles. Figure 3.13 shows that there is indeed a significant difference in friction for different pitch angles. As expected, for larger pitch angles there is also a faster decline in the elevation angle over time.

The challenge now is to extract an approximation of the wind resistance from these experiments. For simplicity, the wind resistance is assumed to be linear with respect to $\dot{\psi}$ (even though it is quadratic). Thus the experiment is described by the following differential equation:

$$J_t\ddot{\psi} = -(k_{friction} + k_{wr}(\theta))\dot{\psi}$$

where k_{wr} corresponds to the wind resistance. The solution of this differential equation is

$$\dot{\psi}(t) = \dot{\psi}(0) \cdot e^{-\frac{k_{friction}+k_{wr}(\theta)}{J_t}t}.$$

As before, this function is fitted into the curves in Figure 3.13. This can be seen in Figure 3.14. This yields:

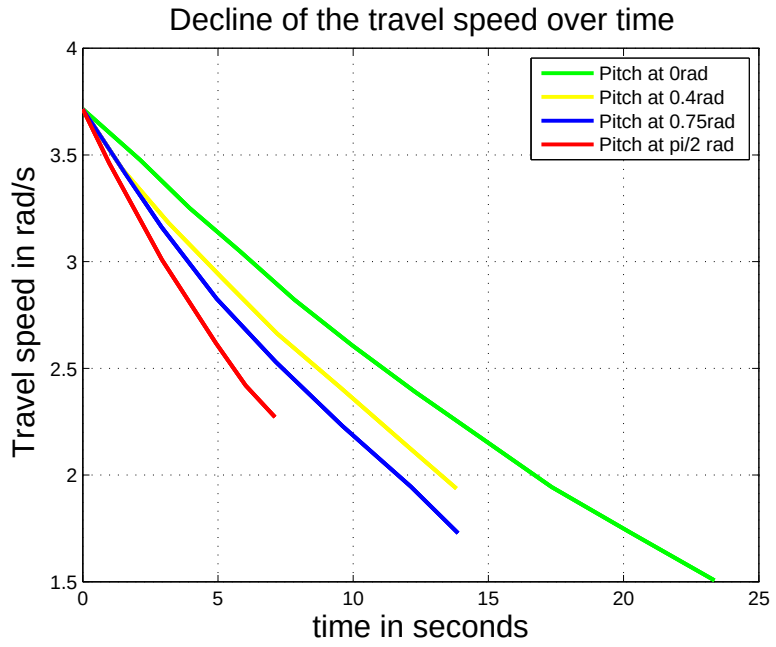

Figure 3.14: Decline of Travel Speed Over Time for Different Pitch Angles, Mean Value.

θ	Function
0	$\dot{\psi}(t) = 3.72e^{-0.03784t}$
0.4	$\dot{\psi}(t) = 3.72e^{-0.04595t}$
0.75	$\dot{\psi}(t) = 3.72e^{-0.05402t}$
$\frac{\pi}{2}$	$\dot{\psi}(t) = 3.72e^{-0.00721t}$

It was assumed that the wind resistance is 0 for $\theta = 0$. As a result, the coefficient k_{wr} can be calculated for the different θ as follows:

θ	k_{wr}
0	0
0.4	$-0.0081\frac{rad}{s}$
0.75	$-0.0168\frac{rad}{s}$
$\frac{\pi}{2}$	$-0.0324\frac{rad}{s}$

Figure 3.15 shows that this can well be approximated by the following linear function:

$$k_{wr}(\theta) = -\,0.02073\theta$$

Consequently the wind resistance is approximated by the following function:

$$F_{wind\psi} = \frac{k_{wr}(\theta)}{d_\psi}\dot{\psi}.$$

This represents the wind resistance around the travel axis. More important is the wind resistance around the elevation axis. When the helicopter is turning very fast around the travel axis, and the controller is using a large pitch angle to slow the helicopter down, the wind resistance will cause lift or downdraft, depending on the direction. This has an significant effect on the controller performance. Therefore it is reasonable to calculate the wind resistance for the elevation axis. This can be done using the geometry depicted in Figure 3.16. There $F_{wind\phi}$ represents the wind resistance on the elevation axis and $F_\perp$ is the wind resistance perpendicular to the helicopters body. Now $F_\perp$ can be derived using trigonometry:

$$\begin{aligned} cos(\theta - \frac{\pi}{2}) = sin(\theta) &= \frac{F_\perp}{F_{wind\psi}} \\ \Rightarrow F_\perp &= F_{wind\psi} sin(\theta). \end{aligned}$$

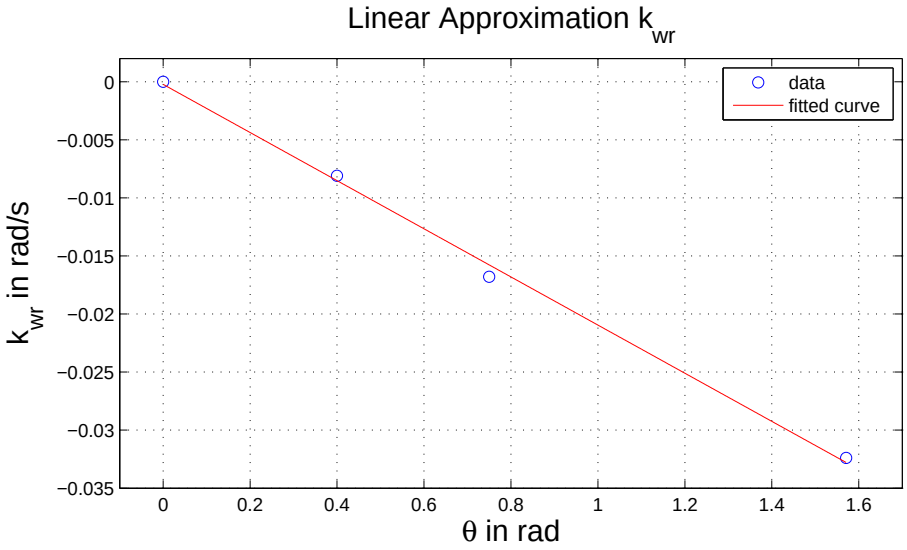

Figure 3.15: Linear Approximation of k_{wr}.

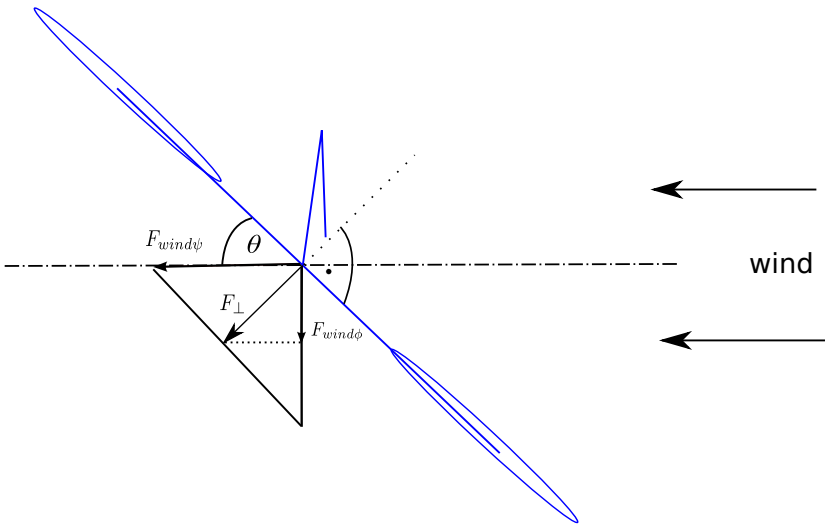

Figure 3.16: Wind Resistance.

In the second step, $F_{wind\phi}$ can now be calculated:

$$\begin{aligned} F_{wind\phi} &= F_{\perp} cos(\theta) \\ &= F_{wind\psi} sin(\theta) cos(\theta) \\ &= \frac{1}{2} F_{wind\psi} sin(2\theta) \\ &= \frac{1}{2} sin(2\theta) \frac{k_{wr}(\theta)}{d_{\psi}} \dot{\psi}. \end{aligned}$$

3.2.6 Summary

$$\begin{aligned} \dot{x}_1 =& x_2 \\ \dot{x}_2 =& \frac{1}{J_p}(d_{\theta w} u_{diff} + \|M_{g\theta}\| + \|M_{cor}\|) \\ \dot{x}_3 =& x_4 \\ \dot{x}_4 =& \frac{1}{J_e}(d_{\phi} cos(x_1) u_{sum} - d_{\phi} cos(x_3)(0.1996 x_3 + 0.03383) kg + \|M_{cen}\| \\ & + \|M_{torque\phi}\| + \|M_{wind\phi}\|) \\ \dot{x}_5 =& x_6 \\ \dot{x}_6 =& \frac{1}{J_t}(d_{\psi} sin(x_1) u_{sum} - k_{friction} x_6 + \|M_{torque\psi}\| + \|M_{wind\psi}\|). \end{aligned}$$

3.3 Model Verification

Before a controller can be implemented, it is essential to validate the model. To do so, several step response experiments were conducted on the real systems as well as the simulation model. The response of model and real plant was the compared graphically. For simplicity, the step response experiment were conducted in such a way that only one of the axis was effected each time.

- Experiment 1: Step on Sum Force with fixed pitch angle $\theta_0 = 0 rad$.
- Experiment 2: Step on Differential Force with fixed travel and elevation angles.
- Experiment 3: Step on Sum Force with fixed pitch angle $\theta_0 = \frac{\pi}{2} rad$.

3.3.0.1 Experiment 1

In the first experiment a step of from $0N$ to $3.2N$ was applied on the sum force. The pitch axis was fixed at $0rad$. Two elements of the model can be validated with this experiment:

1. Elevation Axis
   With the pitch angle fixed at zero, the sum force will entirely act in the direction of the elevation axis. Consequently, the elevation model can be validated using this experiment. Figure 3.17 depicts the result of the experiment for the first three seconds. It can be noted that simulation and experiment show a very similiar step response.

2. Drag Force
   With the pitch angle fixed at zero, the drag momentum created by the rotors is entirely acting in the direction of the travel axis. Consequently, the model of the drag momentum can be validated by comparing the step response on the travel axis with corresponding simulation results. Figure 3.18 shows those results for the first three seconds

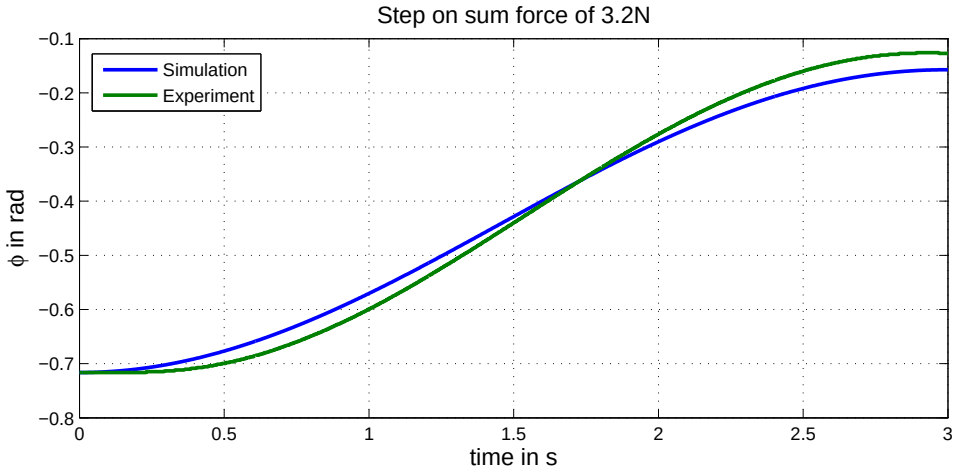

Figure 3.17: Step on Sum Force, Elevation Axis.

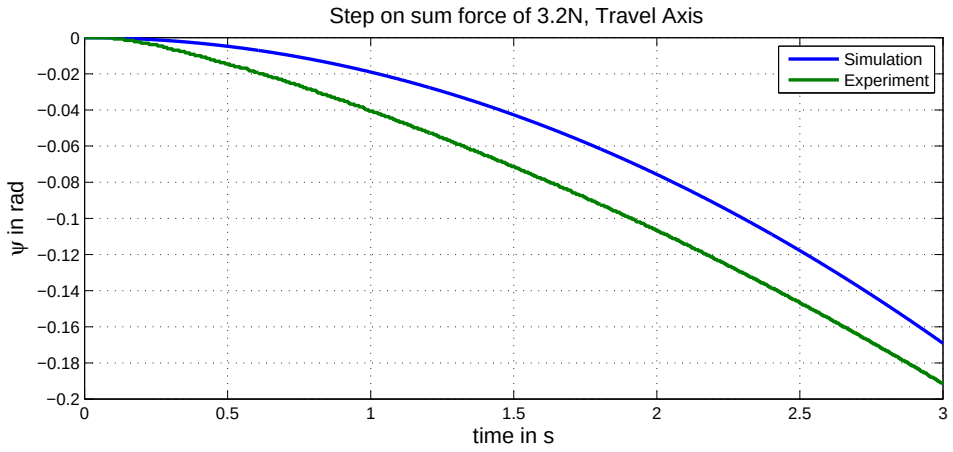

Figure 3.18: Step on Sum Force, Travel Axis.

3.3.0.2 Experiment 2

In the second experiment, the elevation and travel axis were fixed at zero. Then a step of $\Delta u_{diff} = 1.5N$ was applied to the differential force. This experiment has a direct effect on the pitch angle, so it can be used to validate the pitch axis model. The step response on the pitch angle and the corresponding simulation response is depicted in Figure 3.19 for the first second. It shows a good match between simulation and experiment.

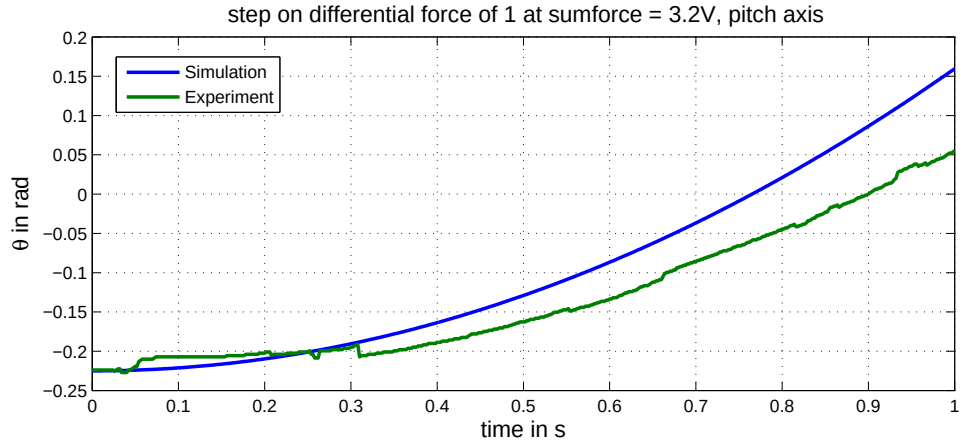

Figure 3.19: Step on Differential Force, Elevation Axis.

3.3.0.3 Experiment 3

In the third experiment, the helicopter was fixed on the pitch axis at $\theta_0 = \frac{\pi}{2}$ and rotated around the travel axis with an initial speed. A step on the sum force was then applied to slow the helicopter down. As the pitch angle is fixed at $\theta_0 = \frac{\pi}{2}$, the sum force is entirely acting on the travel axis. Consequently, this experiment can be used to validate the model of the travel axis. Figure 3.20 depicts the response on the travel axis with the corresponding simulation results for the first five seconds. The step time is $t = 0s$. It can be seen that model and real system show an excellent match.

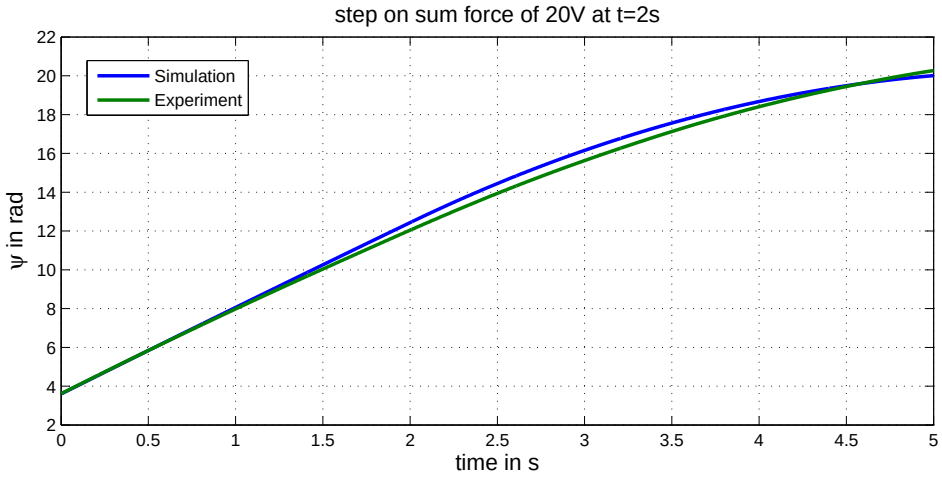

Figure 3.20: Step on Sum Force, Elevation Axis.

4 Controller Design

This chapter will deal with the theory of Model Predictive Control. Additionally, the implementation of the MPC for the 3DOF helicopter is presented and a stability analysis for this controller is discussed.

4.1 Introduction to Model Predictive Control

4.1.1 Introduction

Model Predictive Control (MPC) originates in the process industry and has been used there for the last 30 years. The core idea of MPC is to repeatedly solve an open-loop optimal control problem at each time step. The current state of the system is used as the initial state of the optimal control problem at that time instant. The optimisation problem delivers a sequence of input signal which will steer the system back to its set point - but only the very first input signal is used. The most important advantage of MPC is that input and state constraints can be handled easily, as no explicit control law is needed. This is more difficult and often impossible in traditional infinite horizon optimal control, where an explicit control law has to be determined a priori. This capability enables MPC to run a system close to its constraints, which is often more effective. The most important disadvantage of MPC is that it is computationally extremely demanding, as an optimisation problem has to be solved at every single time step. This is especially problematic if the sampling time is very small, which means that there is not much time to solve the optimal control problem. This is the main reason why MPC had been restricted to applications in the process industry, where the systems are very slow and therefore sampling times very long compared to other applications. This leaves enough time for the optimisation algorithm. Since the computational power has been increasing rapidly throughout the last decades, MPC can now be applied to a wider range of application.

4.1.2 MPC Setup

This thesis is using MPC theory based on state-space models. Let's consider the following state space model

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k), \end{aligned} \tag{4.1}$$

where $x(k) \in \mathcal{X} \in \mathbb{R}^n$, $u(k) \in \mathcal{U} \in \mathbb{R}^m$, $y(k) \in \mathbb{R}^p$ denote the state, input and output respectively.

Here, $\mathcal{X}, \mathcal{U}$ are the state and input constraint sets. They are assumed to be polytopes, i.e. can be described by linear inequalities:

$$\begin{aligned} \mathcal{X} &= \{x(k) \in \mathbb{R}^n \mid Hx(k) \leq h\} \\ \mathcal{U} &= \{u(k) \in \mathbb{R}^n \mid Ru(k) \leq r\}. \end{aligned}$$

Additionally, the system is assumed to be both controllable and observable.

Let's also consider the following cost function:

Definition 1. *Cost function*

$$V(k) = \sum_{j=k}^{k+H_p-1} x^T(j)Qx(j) + \sum_{i=0}^{k+H_c-1} u(i)^T Ru(i) + \sum_{i=0}^{k+H_c-1} \Delta u(i)^T \Delta R \Delta u(i), \quad (4.2)$$

where $H_p \in \mathbb{R}$ denotes the prediction horizon and $H_c \in \mathbb{R}$ the control horizon. $Q \in \mathbb{R}^{n\times n}$, $R \in \mathbb{R}^{m\times m}$ and $\Delta R \in \mathbb{R}^{m\times m}$ are weights to be chosen. Q is penalizing the state deviation, R the magnitude of the input and ΔR changes in the input signal.

Note that non quadratic cost functions can also be used in MPC. Quadratic cost functions are used in this thesis, because they are computationally less demanding, widely used and suitable for the task.

With this definition, the following optimisation problem can be formulated:

Definition 2. *Optimisation problem*

$$\begin{aligned} \min_u \quad & \sum_{j=k}^{k+H_p-1} x^T(j)Qx(j) + \sum_{i=0}^{k+H_c-1} u(i)^T Ru(i) \quad (4.3)\\ \text{subject to } & x(j) \in \mathcal{X} \\ & u(i) \in \mathcal{U} \\ & x(j) = Ax(j-1) + Bu(j-1) \end{aligned}$$

If $H_c < H_p$, then $u(k+H_c) = u(k+H_c+1) = \ldots = u(k+H_p)$ is an additional condition.

With those definitions a basic MPC algorithm can then be formulated as follows:

Definition 3. *Basic MPC algorithm*

1. *Retrieve current measurement $x(k)$*
2. *Determine the next input $u(k)$ by solving the optimisation problem* (4.3)
3. *Apply $u(k)$*
4. *Repeat from 1.*

4.1.3 Offset Free Control

Since the nominal model always contains small errors, there is a difference between the behaviour of the real system and what is predicted by the model. Additionally, unknown

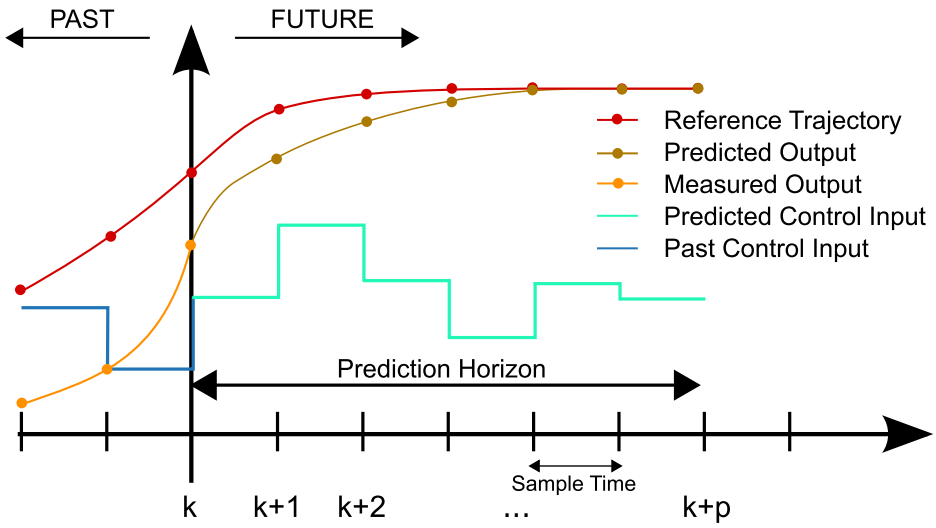

Figure 4.1: Basic MPC Scheme, Picture From [con].

disturbance can increase this model error. In Model Predictive Control this can result in control offsets. In the case of the laboratory helicopter, the relation between the impressed voltage and the rotor's thrust is not perfect. Moreover, nonlinear affects and unknown aerodynamic disturbances add to the model error. Traditional methods mainly deal with this issue by adding integrators to the control law. There are different methods in literature to achieve offset free control for MPC. The Matlab Model Predictive Control Toolbox also contains routines for offset free control. Unfortunately, those routines do not always work properly. Therefore, a self implemented approach is used in this thesis, which is based on the following result presented by [Ekh13] (page 44, Proposition 5.1):

Assume that the steady-state target problem is feasible and that a MPC is implemented using the following augmented model:

Definition 4. *Augmented System*

$$\begin{bmatrix} x \\ d \end{bmatrix}^{+} = \begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix} \begin{bmatrix} x \\ d \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{4.4}$$

$$y = \begin{bmatrix} C & C_d \end{bmatrix} \begin{bmatrix} x \\ d \end{bmatrix} \tag{4.5}$$

According to [Ekh13] the following theorem then applies:

Theorem 1. *Offset Free Control*
Assuming $C_z = HC$, $n_d = m$ and B_d, C_d are chosen such that

$$rank \begin{bmatrix} I - A & -B_d \\ C & C_d \end{bmatrix} = n + m \tag{4.6}$$

Then if the closed-loop system converges to a steady-state with constraints inactive, there is no offset on the controlled outputs.

Proof see [Ekh13] p.45.

Unfortunately, using this augmented model to achieve offset free control will increase the computational complexity compared to a non augmented model, as the augmented model consists of more states. In the case of the laboratory helicopter, this actually is a problem. Because of that a slightly different approach was taken: Instead of using the augmented model in the Model Predictive Controller directly, it was used to estimate the input disturbance using a Lueneberger observer. The input disturbance estimated by the Lueneberger observer is then included into the Model Predictive Controller as a measured disturbance.

4.1.4 Stability

Intuitively, one may expect that the optimality of the MPC algorithm results in guaranteed stability. But in the case of a finite prediction horizon, optimal controllers can actually be unstable, as optimality is only guaranteed for a limited horizon. For linear MPC, as a rule of thumb one can say that a longer horizon makes the controller more likely to be stable.

The input and state constraints render a linear system nonlinear. Consequently stability analysis of constraint Model Predictive Control is based on Lyapunov stability theory. The core idea is to use the cost function as a Lyapunov function to ensure stability of the closed-loop system.

To establish the stability theory, it has been found useful to add several features to the MPC. Firstly, a so called terminal cost $V_f(x)$ is included into the MPC's cost function. The terminal cost is penalising the last step within the prediction horizon:

Definition 5. *Cost function with terminal cost $V_f(x)$*

$$V(k) = \sum_{j=k}^{k+N-1} x^T(j)Qx(j) + \sum_{i=0}^{k+M-1} u(i)^T Ru(i) + \underbrace{x^T(k+N)Q_f x(k+N)}_{V_f(x)}, \tag{4.7}$$

where $Q_f \in \mathbb{R}^{n \times n}$ is the terminal weight.

The second feature to be included into the MPC is the so called terminal constraint set $\mathcal{X}_f$:

Definition 6. *Terminal Constraint*
The terminal constraint set $\mathcal{X}_f$ is a set containing a neighbourhood of the origin the MPC has to enter within its predicition horizon, i.e.

$$x(k+N) \in \mathcal{X}_f \subset \mathcal{X}. \tag{4.8}$$

The last new feature is the terminal controller $\kappa_f(x)$, which is active within the terminal constraint set X_f.

With those features, Lyapunov stability theory makes it possible to derive conditions for the controller's stability. Details of the derivation are not discussed here, but can be found in [Sco00], which provides an excellent review of the matter.

According to [Sco00] the following theorem can be formulated:

Theorem 2. *Stabilising MPC*
An MPC with a terminal weight Q_f and terminal constraint set $\mathcal{X}_f$ is locally stabilising within a feasible set $\mathcal{X}$ if the following conditions hold:

1. *$\mathcal{X}_f \subset \mathcal{X}, \mathcal{X}_f$ closed, $0 \in \mathcal{X}_f$ (state constraint satisfied in $\mathcal{X}_f$)*
2. *$\kappa_f(x) \in \mathcal{U}$, $\forall x \in \mathcal{X}_f$ (control constraints satisfied in $\mathcal{X}_f$)*
3. *$Ax + B\kappa_f(x) \in \mathcal{X}_f$, $\forall x \in \mathcal{X}_f$ (control invariance)*
4. *$V_f(Ax + B\kappa_f(x)) + x^T Q x \leq V_f(x)$ (V_f local Lyaponov function).*

There are plentiful possibilities to satisfy those conditions, depending on the properties of the system. There are different suggestion in literature, depending on if the system is linear or nonlinear, constraint or unconstrained, stable or unstable. The 3DOF helicopter is a linear, but constrained and unstable system. In the following, three methods suitable for the 3DOF helicopter are investigated.

4.1.4.1 Terminal Equality Constraint

The most simple method is to choose the terminal constraint set as the origin, e.g.

$$x(k+N) = 0,$$

thus $X_f = \{0\}$. Consequently, conditions 1.-4. in Theorem 2 are fulfilled trivially, if $\mathcal{X}_f = \{0\}$, $\kappa_f(x) = 0$ and $Q_f = 0$.

The advantage of this method is that it can be implemented very easily. A major problem is feasibility though. Long horizon are needed to steer the system to the origin within the prediction horizon. This might not be possible, because computational power is limited.

4.1.4.2 Terminal Constraint Set (Dual Mode)

In this method, a terminal constraint set X_f and a terminal cost $V_f(x)$ are applied. The idea is that the Model Predictive Controller has to steer the system into the terminal constraint set where a local controller $\kappa_f(x)$ takes over. This is sometimes referred to as Dual Mode Model Predictive Control. Accordingly, the local controller $\kappa_f(x)$ is chosen such that it fulfills conditions 1.-3. in Theorem 2 on the given set X_f and $V_f(x)$ so that condition 4 is fulfilled.

4.1.4.3 Unconstrained Cost Function

The idea of this method is to choose the terminal cost in such a way that the finite horizon MPC becomes equivalent to an infinite horizon controller when the constraints are not active. This results in an excellent performance, because of the good properties of Linear Quadratic Regulator such as robustness. It can be shown that if the terminal cost is chosen as

$$V_f(x) = x^T P x,$$

where P is the solution of the algebraic Riccati equation, the MPC becomes equivalent to an LQR if the constraint are not active. Moreover, the prediction and control horizon have to have the same length. Additionally, the terminal constraint set is defined as the set where the constraints are not active. The MPC is then equivalent to the corresponding LQR in the terminal set and conditions 1.-4. in Theorem 2 are fulfilled (Details and proof see [Ekh13]).

Note that this method is a special case of Dual Mode Model Predictive Control.

As all methods above require reaching the terminal set within the prediction horizon, it is essential to calculate the initial states from which this is possible. This is sometimes referred to as the backwards reachable set or initial admissible set. This set denotes the set within the MPC can be operated.

Definition 7. *Backwards Reachable Set*
The Backwards Reachable Set is defined as

$$\mathcal{X}(\mathcal{X}_f, \mathcal{U}) = \{x_0 \mid \exists (u)_{H_p} \in \mathcal{U} : x_{H_p} \in \mathcal{X}_f\} \tag{4.9}$$

where $(u)_{H_p}$ *is the sequence* $\{u_1, ..., u_{H_p}\}$ *with all* $u_i \in \mathcal{U}$ *and* x_{H_p} *is part of the sequence* $\{x_1, ..., x_{H_p}\}$ *with* $x_{k+1} = f(x_k, u_k)$ $\forall k \in [1, H_p - 1]$.

When calculating the backwards reachable set, it is important to distinguish between Model Predictive Controllers with equidistant prediction and control horizon and MPCs where the prediction horizon is longer than the control horizon. In the latter case, the controller is holding the last value of the input signal within the control horizon for the proceeding prediction horizon.

4.1.4.4 Backwards Reachable Set for $H_p = H_c$

The backwards reachable set $\mathcal{X}$ for $H_p = H_c$ can be calculated using the following simple algorithm suggested by [Löf01]:

Algorithm 1. *Backwards Reachable Set*
If the system A is invertible and the prediction horizon equals the control horizon, then the backwards reachable set can be calculated using the following algorithm:

1. *Generate a cloud of initial points $\mathcal{X} = x_i$ within the terminal set X_f*
2. *Find the backwards reachable points, e.g.*
   $r_{ij} = \{A^{-1}x_i - A^{-1}Bu_j \quad \forall x_i \in \mathcal{X}, u_j \in \mathcal{U}\}$
3. *Generate new set of initial points $\mathcal{X} = \{r_{ij}\} \vee \mathcal{X}$*
4. *Repeat from 2. $N - 1$ times*

This algorithm was implemented as a Matlab function (see Appendix 4.1.4.4). Note that the system matrix has to be invertible for this algorithm. This is the case with the Laboratory Helicopter. Consult [Löf01] for non-invertible system matrices.

4.1.4.5 Backwards Reachable Set for $H_p > H_c$

For the case $H_p > H_c$ the algorithm had to be changed by splitting it into two parts. After the control horizon is reached, the controller will hold the last input value for the rest of the prediction horizon. The first part of the algorithm deals with this part. Therefore Algorithm 1 has to be changed as the input signal has to be hold for the whole reach.

Algorithm 2. *Backwards Reachable Set $H_p > H_c$*
If the system A is invertible and prediction horizon equals control horizon, the backwards reachable set for $H_p > H_c$ can be calculated using the following algorithm:

1. *Generate a cloud of initial points $\mathcal{X} = x_i$ within the terminal set*
2. *Pick one $u_j \in \mathcal{U}$, which has not been used yet*
3. *Find the backwards reachable points, e.g.*
   $r_{ij} = \{A^{-1}x_i - A^{-1}Bu_j \quad \forall x_i \in \mathcal{X}\}$
4. *Repeat 3. $H_p - H_c$ times*
5. *Go to 2. until all u_j have been used*
6. *Generate $\mathcal{X}_1 = \{r_{ij}\} \vee \mathcal{X} \quad \forall i, j$*
7. *Apply $\mathcal{X}_1$ as an initial set in Algorithm 1*

Please consider Appendix 4.1.4.4 for the implementation of this algorithm.

4.1.5 Trajectory Planning

In traditional control methods, trajectory planning can increase the controller's performance significantly. This is due to the fact that the controller may need a different tuning for small deviation from the reference target than for large ones. If a sufficiently smooth trajectory is used, this deviation stays small at all times and the controller can be tuned accordingly.

In theory, model predictive controllers do not necessarily benefit from a trajectory planning, as they already have a type of inherent trajectory planning. Within the model predictive controllers an optimisation algorithm is looking for an input signal which is resulting in a trajectory minimizing the cost function. So in principle, MPC includes trajectory planning. In practise though, very limited prediction horizons restrict these trajectory planning capabilities to deviations from the reference target which lay within the prediction horizon. If the deviation is outside this area, the MPC cannot "see" far enough to plan a good trajectory. As a result of that, model predictive controllers can actually benefit from external trajectory planning, as they keep the deviation from the reference target within the prediction horizon. This is the case with the laboratory helicopter.

The method used here to generate smooth trajectories is based on polynomials.

4.1.5.1 Polynomials

Let's consider the following scenario: The controller is to control a certain output from y_1 to y_2 in $t_{1,2}$ seconds. In that case, the reference signal can have many shapes, as long as it is reaching y_2 within $t_{1,2}$ seconds. The easiest possibility is a simple step. Unfortunately, this simple approach can cause the problems mentioned above. Second order polynomials of the form

$$y(t) = at^2 + bt + c$$

with $y(t_1) = y_1, y(t_2) = y_2$ and $y'(t_1) = y'_1$ are used instead. These ensure sufficiently smooth trajectory. A function has been implemented to generate those trajectories (See Appendix A.1.2).

4.2 Model Preparation

A linear MPC will be used to control the laboratory helicopter. Consequently, the model's nonlinearities and unknowns have to be analysed first. Afterwards a steady state is derived and the model is linearised.

The helicopter model had been identified as

$$\begin{aligned}
\dot{x}_1 =& x_2 \\
\dot{x}_2 =& \frac{1}{J_p}(d_{\theta w} u_{diff} + \|M_{g\theta}\| + \|M_{cor}\|) \\
\dot{x}_3 =& x_4 \\
\dot{x}_4 =& \frac{1}{J_e}(d_\phi cos(x_1) u_{sum} - d_\phi cos(x_3)(0.1996 x_3 + 0.03383) kg + \|M_{cen}\| \\
& + \|M_{torque\phi}\| + \|M_{wind\phi}\|) \\
\dot{x}_5 =& x_6 \\
\dot{x}_6 =& \frac{1}{J_t}(d_\psi sin(x_1) u_{sum} - k_{friction} x_6 + \|M_{torque\psi}\| + \|M_{wind\psi}\|).
\end{aligned}$$

Nonlinearities

Input nonlinearities $\frac{d_\phi}{J_e} cos(x_1) u_{sum}$ and $\frac{d_\phi}{J_e} sin(x_1) u_{sum}$: Those function can only be approximated properly by a linear function, if the system stays close to the set point. Unfortunately, this is not the case. If you choose e.g. $x_{1s} = 0$ the linear function will just be zero, which is certainly not true when $x_1 = 45°$. This will be dealt with later by modeling these nonlinearities as measured disturbances in the MPC algorithm.

Unknowns

Some effects could not be identified properly and are unknown or only partially known.

- $\|M_{g\theta}\|$: Gravitational force for large pitch angles, because the joint is below the center of gravity.
- $\|M_{cor}\|$: Coriolis force, which is pushing the pitch angle back to zero when rotating around the travel axis. Experiments have shown that it can be very significant for large travel speeds.
- $\|M_{cen}\|$: Centrifugal force which could only be partially identified.
- $\|M_{wind}\|$: Drag caused by wind resistance.

The unknowns will be neglected in the model used in the Model Predictive Controller. Even tough, they will be dealt with by input disturbance estimators to avoid zero-offsets.

The model then simplifies to

$$\begin{aligned}
\dot{x}_1 =& x_2 \\
\dot{x}_2 =& \frac{1}{J_p} d_{\theta w} u_{diff} \\
\dot{x}_3 =& x_4 \\
\dot{x}_4 =& \frac{1}{J_e} (d_\phi cos(x_1) u_{sum} - d_\phi cos(x_3)(0.1996 x_3 + 0.03383) kg \\
\dot{x}_5 =& x_6 \\
\dot{x}_6 =& \frac{1}{J_t} (d_\psi sin(x_1) u_{sum} - k_{friction} x_6).
\end{aligned} \tag{4.10}$$

4.2.1 Steady State

It is intuitively clear that there are many steady states, as the helicopter can hover on any elevation angle and on any travel angle. Therefore, the steady state values those two angles arbitrary and have to be chosen. But let's look at Equation (4.10) to show that.

For the steady state, the left hand side of Equation (4.10) has to be zero. Thus,

$$\begin{aligned}
0 =& x_2 \\
0 =& \frac{1}{J_p} d_{\theta w} u_{diff} \\
0 =& x_4 \\
0 =& \frac{1}{J_e}(d_\phi cos(x_1) u_{sum} - d_\phi cos(x_3)(0.1996 x_3 + 0.03383) kg \\
0 =& x_6 \\
0 =& \frac{1}{J_t}(d_\psi sin(x_1) u_{sum} - k_{friction} x_6).
\end{aligned}$$

Obviously, $x_{2s} = x_{4s} = x_{6s} = u_{diffs} = 0$.

So the following equations are left to be solved:

$$\begin{aligned}
0 &= \frac{1}{J_e}(d_\phi cos(x_{1s}) u_{sums} - d_\phi cos(x_{3s})(0.1996 x_{3s} + 0.03383) kg + \frac{1}{2} sin(x_{1s}) k_{mdrag} u_{sums}^2) \\
0 &= \frac{1}{J_t}(d_\psi sin(x_{1s}) u_{sums} + \frac{1}{2} cos(x_{1s}) k_{mdrag} u_{sums}^2)
\end{aligned} \tag{4.11}$$

which are two equations with the 3 unknowns x_{1s}, x_{3s} and u_{sums}. As a result, one of those unknowns has to be chosen. A reasonable operating point for the helicopter is to hover at an elevation angle of $x_3 = -0.2$. Consequently, x_{3s} is chosen to be $x_{3s} = -0.2$. Note that x_{5s} does not appear in Equation (4.11) and can therefore be chosen arbitrarily. It is chosen to be $x_{5s} = 0$. Now equation (4.11) could be solved numerically. However, there is a much simpler approach: the experiment used to determine the torque in the previous chapter consisted of letting the helicopter hover for different elevation angles. This experiment showed that for $x_{3s} = -0.2$ the sum force is $u_{sums} = 1.3N$ and the pitch angle $x_{1s} = 0.03rad$ when hovering.

Variable	Steady State Value
u_{diff}	$0rad$
u_{sum}	$1.3N$
x_1	$0.03rad$
x_2	$0rad$
x_3	$-0.2rad$
x_4	$0rad$
x_5	$0rad$
x_6	$0rad$

4.2.2 Linearisation

Linearisation at the above steady state yields the following state space model:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ -\frac{1}{J_e} d_\phi sin(x_{1s}) u_{sums} & 0 & \frac{\partial}{\partial x_3} \frac{1}{J_e} M_{g\phi} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{1}{J_t} d_\psi cos(x_{1s}) u_{sums} & 0 & 0 & 0 & 0 & -\frac{1}{J_p} k_{friction} \end{bmatrix}$$

and

$$B = \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{J_p} d_{\theta w} \\ 0 & 0 \\ \frac{1}{J_e} d_\phi cos(x_{1s}) & 0 \\ 0 & 0 \\ \frac{1}{J_t} d_\psi sin(x_{1s}) & 0 \end{bmatrix}$$

or numerically

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ -1.283 & 0 & -0.11 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0.1146 & 0 & 0 & 0 & 0 & -0.0247 \end{bmatrix}$$

and

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0.248 \\ 0 & 0 \\ 0.655 & 0 \\ 0 & 0 \\ 0.00866 & 0 \end{bmatrix}.$$

4.3 Constraints

4.3.0.1 Input Constraints

Each motor can produce a maximum thrust of about $2.2N$ and a minimum thrust of $0N$. So no negative thrust is possible. The MPC is using the sum force u_{sum} and the differential force u_{diff} to control the helicopter. Intuitively, it makes sense to choose $0N \leq u_{sum} \leq 4.4N$. This is not a good choice though, because the range of the differential force u_{diff} actually depends on the current sum force u_{sum}: the idea of the differential force is to increase the thrust of one motor and decrease the thrust of the other motor for the same amount. This way the sum force stays constant when doing so. The problem is that the thrust of each motor is constraint between $0N$ and $2.2N$. So if for example the sum force is currently $u_{sum} = 0N$, the thrust of either motor cannot be reduced any more (because both are at their minimum thrust). Consequently, as long as $u_{sum} = 0N$ the range of u_{sum} shrinks to zero and the pitch axis cannot be manipulated any more. As a result, the range of the sum force has to be reduced, so that a reasonable range for the pitch axis controller remains. A good choice turned out to be a maximum differential force of $u_{diff} = \pm 1N$ and a maximum differential force of $u_{diff} = \pm 1.5N$. This dynamic constraint can be implemented with the Matlab MPC toolbox. Consequently, the sum force has to be restricted to $0.5N \leq u_{sum} \leq 3.9N$.

Table 4.1: Input Constraints.

Variable	Min	Max
u_{sum}	0.5N	3.9N
u_{diff}	-1.5N or $\geq -u_{sum}$	1.5N

4.3.0.2 State Constraints

Both the pitch as well as the elevation angle are constraint to a certain range. The travel angle on the other hand is unconstrained. Physically, the helicopter can move from a pitch angle of -90° up to angle of 90°. But it certainly does not make sense to use this whole range. First of all there has to be a certain margin of error to robustify the controller. So the controller should not be operated at its very physical limit. Moreover, if the pitch angle is close to 90°, all the thrust will be directed around the travel axis and none on the elevation axis. So the controller will not operate the plant at this angle anyway. In the course of setting up the controller, it turned out that $-\frac{\pi}{4} \leq \theta \leq \frac{\pi}{4}$ is a good choice. The elevation angle is physically restricted by a minimum angle when the helicopter is on the floor and a maximum angle. The helicopter is on the floor for $\phi_{min} = -0.7164rad$. As a maximum angle, $\phi_{max} = 0.2rad$ was chosen.

Table 4.2: State Constraints.

Variable	Min	Max
θ	$-\frac{\pi}{4}rad$	$\frac{\pi}{4}rad$
ϕ	$-0.7164rad$	$0.2rad$

4.4 Controller Cascation

The standard approach would be to use the linear model as it is and design a MPC directly. But it turned out that the MPC is very difficult to tune this way, because there are a lot of states - 9 - to consider. Moreover, the high order of the system is computationally very demanding. Because of that it is more practical to cascade the controller. This has the advantage that independent controllers with less states can be tuned individually instead of tuning all at the same time. This is also reducing computational complexity. Additionally, different sampling rate can be used for the individual controllers.

The linearised model can be divided into two loops: All axis are powered by the same rotors. This means that the input forces are all in the same range. If we ignore the cross-connections between pitch and the other axis, the inertias give an idea about the time constants of each axis. The inertias are

$$\begin{aligned} J_p &= 0.0398kgm^2 \\ J_e &= 1.1555kgm^2 \\ J_t &= 1.1785kgm^2. \end{aligned}$$

So the time constant of the pitch axis is around 25 times lower than the time constants of the elevation and travel axis. As a result the pitch axis serves as an inner loop and the travel and elevation axis as an outer loop. As a result, the system equations can be reformulated in the following manner:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0.248 \end{bmatrix} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix}$$

$$\begin{bmatrix} \dot{x_3} \\ \dot{x_4} \\ \dot{x_5} \\ \dot{x_6} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1.283 & -0.11 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -0.0247 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0.655 & 0 \\ 0 & 0 \\ 0.00866 & 0 \end{bmatrix} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 0.00076 \\ 0 \\ 0.1146 \end{bmatrix} x_1}_{F_{x1}}$$

where F_{x1} is used to model the influence of x_1 on elevation and pitch as a measured disturbance.

You can also see, that elevation and pitch angle are independent now. So the system can actually be split into 3 individual 2x2 systems:

$$\begin{aligned}
\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A_{pitch}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0.248 \end{bmatrix}}_{B_{pitch}} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} \\
\begin{bmatrix} \dot{x_3} \\ \dot{x_4} \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & -0.11 \end{bmatrix}}_{A_{elev}} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0.655 & 0 \end{bmatrix}}_{B_{elev}} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} + \begin{bmatrix} 0 \\ -1.283 \end{bmatrix} x_1 \\
\begin{bmatrix} \dot{x_5} \\ \dot{x_6} \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & -0.0247 \end{bmatrix}}_{A_{travel}} \begin{bmatrix} x_5 \\ x_6 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0.00866 & 0 \end{bmatrix}}_{B_{travel}} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.1146 \end{bmatrix} x_1.
\end{aligned} \tag{4.12}$$

This restructured model results in the controller structure depicted in Figure 4.2. There, the elevation and travel controller form two parallel independent outer loops. They are provided with the reference trajectories x_{3ref} for the elevation angle and x_{5ref} for the travel angle respectively. The elevation controller provides the sum force u_{sum} as an input to the helicopter. The travel controller on the other hand does not provide a direct input signal to the helicopter, but a reference trajectory x_{1ref} for the pitch angle. This reference trajectory x_{1ref} is processed by the pitch controller, which forms an inner loop with respect to the travel controller. The pitch controller then provides the differential force u_{diff} as an input signal to the helicopter.

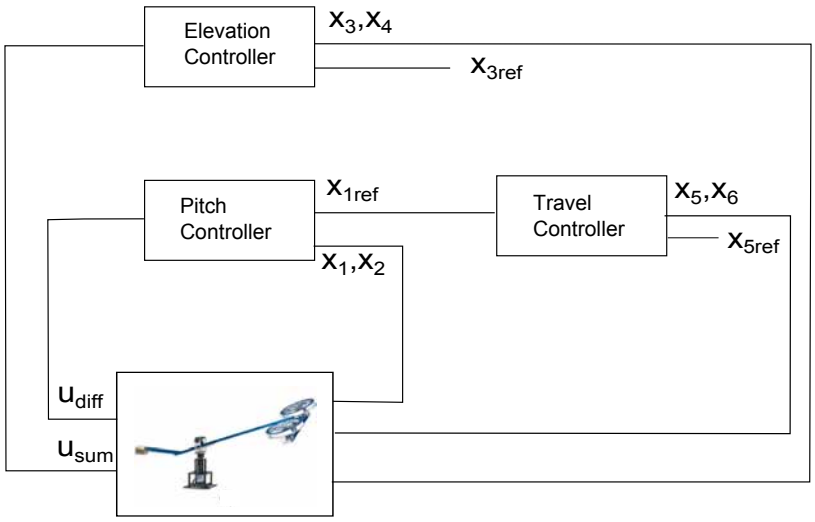

Figure 4.2: Controller Cascation Structure.

4.5 Nonlinear Aspects

A linear Model Predictive Controller can only use a linear model and all nonlinear affects have to be neglected. In the case of the Quanser 3DOF helicopter there are some significant nonlinear aspects though. The most important nonlinearity are the input nonlinearities on the elevation and travel axis. If one considers the original nonlinear model

$$
\begin{aligned}
\dot{x}_1 =& x_2 \\
\dot{x}_2 =& \frac{1}{J_p}(d_{\theta w} u_{diff} + \|M_{g\theta}\| + \|M_{cor}\|) \\
\dot{x}_3 =& x_4 \\
\dot{x}_4 =& \underbrace{\frac{1}{J_e}(d_\phi cos(x_1) u_{sum}}_{B_{elev,nonlin}(x_1,u_{sum})} - d_\phi cos(x_3)(0.1996 x_3 + 0..03383)kg + \|M_{cen}\| \\
& + \|M_{torque\phi}\| + \|M_{wind\phi}\|) \\
\dot{x}_5 =& x_6 \\
\dot{x}_6 =& \underbrace{\frac{1}{J_t}(d_\psi sin(x_1) u_{sum}}_{B_{travel,nonlin}(x_1,u_{sum})} - k_{friction} x_6 + \|M_{torque\psi}\| + \|M_{wind\psi}\|)
\end{aligned}
$$

those input nonlinearities can be identified as

$$B_{elev,nonlin}(x_1, u_{sum}) = \frac{d_\phi}{J_e} cos(x_1) u_{sum}$$
$$B_{travel,nonlin}(x_1, u_{sum}) = \frac{d_\psi}{J_t} sin(x_1) u_{sum}.$$

By adding and subtracting $\frac{d_\phi}{J_e} cos(x_{1s}) u_{sum}$, those nonlinearities can be reformulated as

$$B_{elev,nonlin}(x_1, u_{sum}) = \underbrace{\frac{d_\phi}{J_e} cos(x_{1s})}_{B_{elev}[2,1]} u_{sum} + \underbrace{(cos(x_1) - cos(x_{1s})) \frac{d_\phi}{J_e} u_{sum}}_{\delta_{elev}(x_1, u_{sum})}$$
$$B_{travel,nonlin}(x_1, u_{sum}) = \underbrace{\frac{d_\psi}{J_t} sin(x_{1s})}_{B_{travel}[2,1]} u_{sum} + \underbrace{(sin(x_1) - sin(x_{1s})) \frac{d_\psi}{J_t} u_{sum}}_{\delta_{travel}(x_1, u_{sum})},$$

where $B_{elev}[2,1]$ and $B_{travel}[2,1]$ refer to the second element in the first column of B_{elev} and B_{travel} respectively (refer to Equation (4.12)).

Consequently, $\delta_{elev}(x_1, u_{sum})$ and $\delta_{travel}(x_1, u_{sum})$ can be added to Equation (4.12) as an input disturbance. This yields

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0.248 \end{bmatrix} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix}$$
$$\begin{bmatrix} \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1.283 & -0.11 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0.655 & 0 \end{bmatrix} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} + \begin{bmatrix} 0 \\ -1.283 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \delta_{elev}(x_1, u_{sum})$$
$$\begin{bmatrix} \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -0.0247 \end{bmatrix} \begin{bmatrix} x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0.00866 & 0 \end{bmatrix} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.1146 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \delta_{travel}(x_1, u_{sum}).$$

Another important non-linearity is the centrifugal force. For the set point $x_{6s} = 0$ the centrifugal force is zero in the linear model. Unfortunately, the helicopter is also operated on $x_6 \ll 0$. To counteract this, another measured disturbance is introduced:

$$\delta_{cen}(x_6) = M_{cen\phi}(\phi)$$

where $M_{cen\phi}(\phi)$ has been identified in the previous chapter.

As a result, the model expands to

$$\begin{aligned}
\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} &= \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0.248\end{bmatrix}\begin{bmatrix}u_{sum}\\ u_{diff}\end{bmatrix}\\
\begin{bmatrix}\dot{x_3}\\ \dot{x_4}\end{bmatrix} &= \begin{bmatrix}0 & 1\\ -1.283 & -0.11\end{bmatrix}\begin{bmatrix}x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0.655 & 0\end{bmatrix}\begin{bmatrix}u_{sum}\\ u_{diff}\end{bmatrix}\\
&\quad \begin{bmatrix}0\\ -1.283\end{bmatrix} x_1 + \begin{bmatrix}0\\ 1\end{bmatrix}(\delta_{elev}(x_1, u_{sum}) + \delta_{cen}(x_6))\\
\begin{bmatrix}\dot{x_5}\\ \dot{x_6}\end{bmatrix} &= \begin{bmatrix}0 & 1\\ 0 & -0.0247\end{bmatrix}\begin{bmatrix}x_5\\ x_6\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0.00866 & 0\end{bmatrix}\begin{bmatrix}u_{sum}\\ u_{diff}\end{bmatrix}\\
&\quad + \begin{bmatrix}0\\ 0.1146\end{bmatrix} x_1 + \begin{bmatrix}0\\ 1\end{bmatrix}\delta_{travel}(x_1, u_{sum}).
\end{aligned} \tag{4.13}$$

4.6 Pitch Controller

The pitch axis was described by the following model

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0.248\end{bmatrix}\begin{bmatrix}u_{sum}\\ u_{diff}\end{bmatrix}$$

or simpler

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ 0.248\end{bmatrix} u_{diff}$$

As mentioned in Section 4.2, there are also unknowns in the system, which have to be taken care of by the controller. Those are

1. Coriolis Force
2. Gravitational Momentum.

Additionally, an input disturbance will be assumed.

Formally, these can be described as

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ 0.248\end{bmatrix} u_{diff} + \begin{bmatrix}0\\ 1\end{bmatrix} \|M_{Cor}\| + \begin{bmatrix}0\\ 1\end{bmatrix} \|M_{g\theta}\| + \begin{bmatrix}0\\ 0.248\end{bmatrix} \|\triangle u_{diff}\|$$

$$\Leftrightarrow \begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ 0.248\end{bmatrix} u_{diff} + \begin{bmatrix}0 & 0\\ 1 & 0.248\end{bmatrix} \begin{bmatrix}\|M_{Cor}\| + \|M_{g\theta}\|\\ \|\triangle u_{diff}\|\end{bmatrix}$$

or

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}}_{A_p}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \underbrace{\begin{bmatrix}0\\ 0.248\end{bmatrix}}_{B_p} u_{diff} + \underbrace{\begin{bmatrix}0\\ 1\end{bmatrix}}_{B_{p\delta}} \underbrace{(\|M_{Cor}\| + \|M_{g\theta}\| + 0.248\|\triangle u_{diff}\|)}_{\delta_{inpitch}} \quad (4.14)$$

So next to meeting the computational limits, the controller has also be able to manage the input disturbances on the pitch axis.

4.6.1 Sampling Rate

On the one hand, the sampling rate on the pitch axis should be as large as possible. This has two positive affects on the prediction and control horizons:

1. The longer the sampling interval, the longer the time frame captured by the MPC.

2. If the sampling interval is longer, there is more time to solve the optimisation problem and therefore longer prediction and control horizons can be used.

On the other hand the sampling rate has to be small enough to capture all relevant dynamics. Trial and error has shown that a good rule of thumb is to take the cutoff frequency and multiply it with 40. The bode plot of the pitch axis is

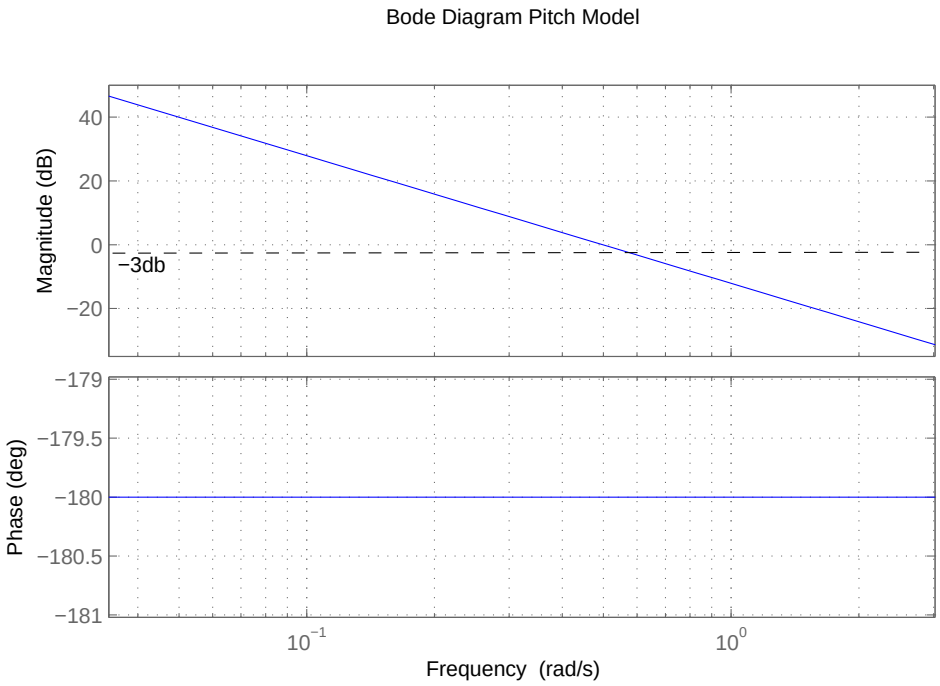

Figure 4.3: Bode Plot Pitch Axis.

The cutoff frequency at -3dB is $10^{0.472}Hz$. So the sampling frequency is $f_{sp} = 40 \cdot 10^{0.472}Hz = 120Hz$. This corresponds to a sampling time of about

$$T_{spmax} = 0.0083s \approx 0.008s.$$

4.6.2 Controller Parameters

4.6.2.1 Prediction and Control Horizon

From a theoretical point of view, prediction and control should be as long as possible. This ensures good stability properties and makes the stability proof much easier. In practise though, these horizons are limited by the available computational power and the sampling rate. So finding liable horizons is an incremental process, where the horizons are increased until the computational limit is reached. Important in that process is the fact that long prediction horizons are much cheaper than long control horizons. So much longer prediction horizons than control horizon can be a better choice than even ones.

It turned that a prediction horizon of

$$H_p = 80$$

is a good compromise. This corresponds to a prediction length of $0.64s$.

The performance was practically identical for long control horizons compared to small control horizons. Consequentially, a very small control horizon of

$$H_c = 4$$

was chosen.

4.6.2.2 Weights

The state weight Q, the input weights R and ΔR has been determined incrementally by trial and error to achieve good performance:

Parameter	Value
Q_{pitch}	$\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$
R_{pitch}	0.01
$\triangle R_{pitch}$	$1 \cdot 10^{-5}$

4.6.3 Input Disturbance Estimation

To achieve offset free control and to enhance the controller's performance, an input disturbance estimator is added to the controller. The estimated disturbance will be subtracted from the input signal. In (4.14) the input disturbance was described by

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A_p} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 0.248 \end{bmatrix}}_{Bp} u_{diff} + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B_{pd}} d_{inpitch}. \tag{4.15}$$

To design a input disturbance estimator, the system has to be augmented in the following way:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{d}(t) \end{bmatrix} = \begin{bmatrix} Ap & B_{pd} \\ 0 & I \end{bmatrix} \begin{bmatrix} x(t) \\ d(t) \end{bmatrix} + \begin{bmatrix} B_p \\ 0 \end{bmatrix} u$$

$$y(t) = \begin{bmatrix} C & C_{pd} \end{bmatrix} \begin{bmatrix} x(t) \\ d(t) \end{bmatrix}.$$

Above it was shown that the disturbance can be estimated if

$$rank \begin{bmatrix} I - Ap & -B_{pd} \\ C & C_{pd} \end{bmatrix} = n + m$$

where $n = 2$ is the dimension of A_p and $m = 1$ the number of disturbances to be estimated.

This is the case.

Using the augmented system, a Lueneberger Observer was designed. Critical is the speed of the observer, because the observer should not interfere with the dynamics of the controller. As a result, the observer has to be significant slower than the controller. A good compromise turned out to be the following choice of poles for the discrete observer:

$$\text{Poles} = [0, 0, 0.999].$$

4.7 Elevation Controller

The elevation axis was described by

$$\begin{bmatrix} \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1.283 & -0.11 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0.655 & 0 \end{bmatrix} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (\delta_{elev}(x_1, u_{sum}) + \delta_{cen}(x_6)).$$

Additionally the following disturbances are taken into account

1. Offset in the gravitational momentum
2. Input disturbance.

This results in the following extended model:

$$\begin{bmatrix} \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1.283 & -0.11 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0.655 & 0 \end{bmatrix} \begin{bmatrix} u_{sum} \\ u_{diff} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (\delta_{elev}(x_1, u_{sum}) + \delta_{cen}(x_6)) + \underbrace{\begin{bmatrix} 0 \\ 0.248 \end{bmatrix} \triangle u_{sum} + \triangle M_{g\phi}}_{\delta_{ud}(u_{sum}, x_5)}$$

or simpler

$$\begin{bmatrix} \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1.283 & -0.11 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0.655 \end{bmatrix} u_{sum} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (\delta_{elev}(x_1, u_{sum}) + \delta_{cen}(x_6)) + \delta_{ud}(u_{sum,}x_5).$$

4.7.1 Sampling Rate

As for the pitch axis, the cutoff frequency multiplied with 40 is used as the sampling rate for the elevation axis. The bode plot of the elevation axis is

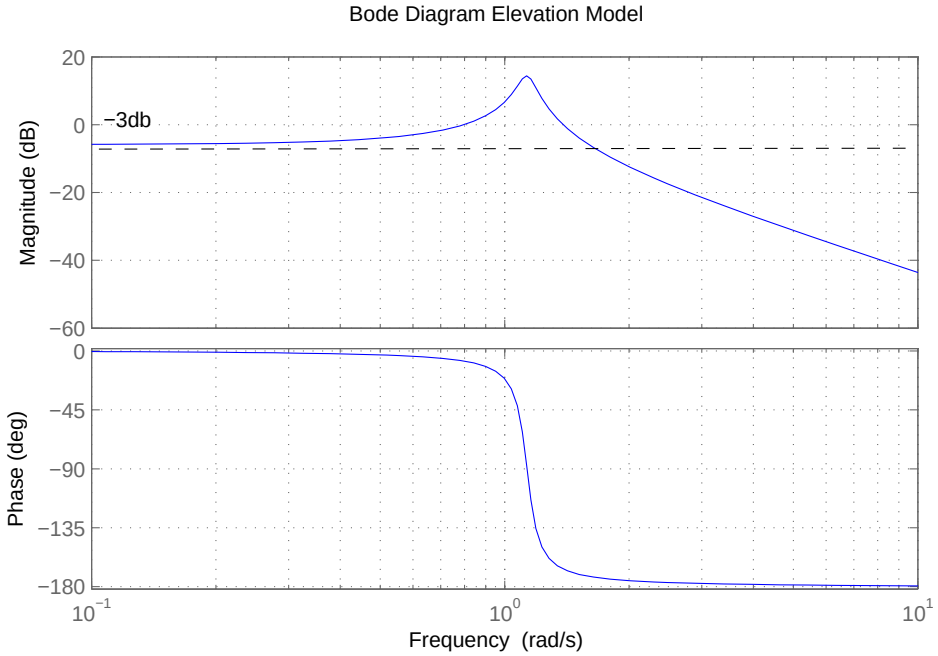

Figure 4.4: Bode Plot Elevation Axis.

The cutoff frequency at -3dB can be read out as approximately $10^{0.1}Hz$. So the sampling frequency is $f_{sp} = 40 \cdot 10^{0.1}Hz = 50.35Hz$ and thus

$$T_{semax} = 0.02s.$$

Unfortunately, this sampling frequency could not be used in the experiments. Most likely because of a bug in the MPC toolbox, the trajectory tracking was not working for sampling rates not equal to 0.002s, which is the sensor sampling rate. For this reason, the sampling rate $T = 0.002s$ will be used in the following.

4.7.2 Controller Parameters

Prediction and Control Horizon

Parameter	Value
Prediction Horizon	15
Control Horizon	5

Note that prediction and control horizon are significantly shorter for the elevation controller than for the other axis. This is due to the fact that the sampling interval of $T = 0.002s$ on the elevation axis is much shorter than on the other axis. That leaves less time for the algorithm to solve the optimisation problem. In consequence, the horizons have to be shorter to solve the optimisation problem in a shorter amount of time.

Weights

Parameter	Weight
Q_{elev}	$1000 \cdot \begin{bmatrix} 40 & 0 \\ 0 & 4 \end{bmatrix}$
R_{elev}	0.0001
$\triangle R_{elev}$	0.01

4.7.3 Input Disturbance Estimation

Model Predictive Controllers cannot deal with input disturbances directly, as they do not have an inherit "integral" part as e.g. PI-controllers. As a result, the input disturbance will cause an output offset. To achieve zero offset, a input disturbance estimator is added to the controller. The estimated disturbance will be subtracted from the input signal. In (4.15) the input disturbance where described as

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A_p} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 0.248 \end{bmatrix}}_{Bp} u_{diff} + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B_{pd}} d_{inpitch}. \tag{4.16}$$

To design a input disturbance estimator, the system has to be augmented in the following way:

$$\begin{bmatrix}\dot{x}(t)\\ \dot{d}(t)\end{bmatrix} = \begin{bmatrix}A_e & B_{ed}\\ 0 & I\end{bmatrix}\begin{bmatrix}x(t)\\ d(t)\end{bmatrix} + \begin{bmatrix}B_e\\ 0\end{bmatrix} u$$
$$y(t) = \begin{bmatrix}C & C_{ed}\end{bmatrix}\begin{bmatrix}x(t)\\ d(t)\end{bmatrix}.$$

Using the augmented system, a Lueneberger Observer was designed. A good compromise turned out to be the following choice of poles for the discretised observer:

$$\text{Poles} = [0, 0, 0.9995].$$

4.7.4 Pitch Axis Compensation

As described before, in the linearisation it has to be assumed that the states operate in a small range. Unfortunately, this is not the case for the pitch angle. Therefore, the generated thrust in elevation direction has an offset for pitch angles not equal to zero. To increase the performance of the controller, a measured disturbance as a compensation is introduced. The measured disturbance is defined as

$$\delta_{elev}(x_1, u_{sum}) = (cos(x_1) - cos(x_{1s}))d_\phi u_{sum}.$$

4.8 Travel Controller

The travel axis was described by the following model

$$\begin{bmatrix}\dot{x}_5\\ \dot{x}_6\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & -0.0247\end{bmatrix}\begin{bmatrix}x_5\\ x_6\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0.00866 & 0\end{bmatrix}\begin{bmatrix}u_{sum}\\ u_{diff}\end{bmatrix} + \begin{bmatrix}0\\ 0.1146\end{bmatrix} x_1 + \begin{bmatrix}0\\ 1\end{bmatrix}\delta_{travel}(x_1, u_{sum})$$

or simpler

$$\begin{bmatrix}\dot{x}_5\\ \dot{x}_6\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & -0.0247\end{bmatrix}\begin{bmatrix}x_5\\ x_6\end{bmatrix} + \begin{bmatrix}0\\ 0.00866\end{bmatrix} u_{sum} + \begin{bmatrix}0\\ 0.1146\end{bmatrix} x_1 + \begin{bmatrix}0\\ 1\end{bmatrix}\delta_{travel}(x_1, u_{sum}).$$

Additionally several disturbances have to be taken into account:

1. Disturbance on Sum Force

2. Input Disturbance.

Formally, these can be described as

$$\begin{bmatrix}\dot{x}_5\\ \dot{x}_6\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_5\\ x_6\end{bmatrix} + \begin{bmatrix}0\\ 0.248\end{bmatrix} u_{diff} + \begin{bmatrix}0\\ 1\end{bmatrix} \|M_{Cor}\| + \begin{bmatrix}0\\ 0.248\end{bmatrix} \|\triangle u_{diff}\| + \begin{bmatrix}0\\ 1\end{bmatrix} \delta_{travel}(x_1, u_{sum})$$

or

$$\begin{bmatrix}\dot{x}_5\\ \dot{x}_6\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_5\\ x_6\end{bmatrix} + \begin{bmatrix}0\\ 0.248\end{bmatrix} u_{diff} + \begin{bmatrix}0 & 0\\ 1 & 0.248\end{bmatrix} \begin{bmatrix}\|M_{Cor}\| + \delta_{travel}(x_1, u_{sum})\\ \|\triangle u_{diff}\|\end{bmatrix}$$

or

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ 0.248\end{bmatrix} u_{diff} + \underbrace{\begin{bmatrix}0\\ 1\end{bmatrix}}_{B_{t\delta}} \underbrace{(\|M_{Cor}\| + 0.248\|\triangle u_{diff}\| + \delta_{travel}(x_1, u_{sum}))}_{\delta_{inpitch}}. \tag{4.17}$$

4.8.1 Sampling Rate

As before, the cutoff frequency multiplied with 40 is used as the sampling rate for the travel axis. The bode plot of the travel axis is shown in Figure 4.5.

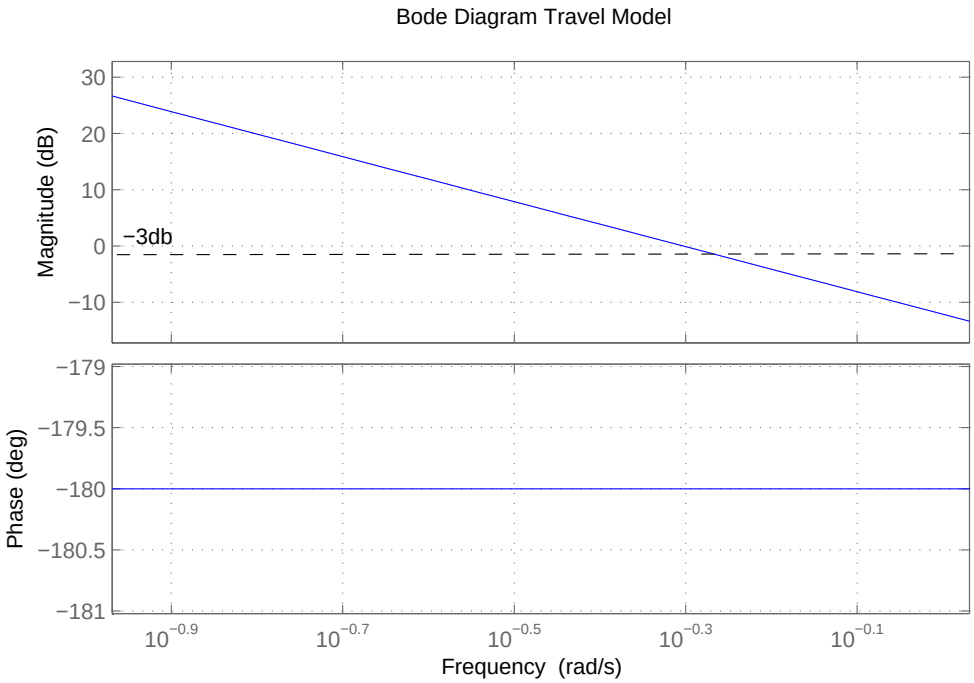

Figure 4.5: Bode Plot Travel Axis.

The cutoff frequency at -3dB can be read out as $10^{-0.33}Hz$. So the sampling frequency is $f_{sp} = 40 \cdot 10^{-0.33}Hz = 18.7Hz$ and thus

$$T_{stmax} = 0.053s.$$

It turned out though that this sampling interval is actually too short and the controller's performance was affected in consequence. Thus the sampling interval was chosen to be the same as for the pitch axis:

$$T_{stmax} = 0.008s.$$

4.8.2 Controller Parameters

Prediction and Control Horizon

Parameter	Value
Prediction Horizon	80
Control Horizon	4

Weights

Parameter	Weight
Q_{travel}	$\begin{bmatrix} 0.04 & 0 \\ 0 & 0.015 \end{bmatrix}$
R_{travel}	$1 \cdot 10^{-6}$
$\triangle R_{travel}$	$1 \cdot 10^{-5}$

4.9 Nominal Stability

The Model Predictive Controller implemented above is working and does achieve an excellent performance. There is no structural guarantee though that this controller is actually stable for all possible trajectories. To design a Model Predictive Controller with guaranteed nominal stability, MPC stability theory has to be applied. As previously mentioned in 4.1.4, three different possibilities of satisfying the conditions for nominal stability are investigated in this thesis. Those are

- the MPC with an Unconstrained Cost,
- the Dual Mode Model Predictive Control,
- the Terminal State Method.

The main issue with nominal stability in practise is a too small feasible sets in which the controller can be operated. The feasible set has to be large enough to be able to operate the controller in all possible scenarios. Unfortunately, it is not so clear how large this feasible set actually has to be. It does not necessarily have to be to whole state space. The important factor is the deviation from the given reference target $\Delta x = x - x_{ref}$ as the feasible set describes the tolerable deviation from the reference target. How large the tolerable deviation can be depends for example on the expected disturbances, model errors and the reference trajectory. In consequence, a larger feasible set should result in a better robustness of the controller. If the reference trajectory includes fast changes, the terminal constraint set should also be large.

In the case of the laboratory helicopter, it is reasonable to assume a significant model error (as shown in section 2) as well as some disturbances. The reference trajectories used in simulation and experiments later will also be rather fast. Therefore the terminal set should have a certain size. Consequently, the following specification are made for the terminal set:

Variable	Min	Max
Δx_1	$-0.5rad$	$0.5rad$
Δx_2	$-0.25\frac{rad}{s}$	$0.25\frac{rad}{s}$
Δx_3	$-0.5rad$	$0.5rad$
Δx_4	$-0.25\frac{rad}{s}$	$0.25\frac{rad}{s}$
Δx_5	$-0.5rad$	$0.5rad$
Δx_6	$-0.25\frac{rad}{s}$	$0.25\frac{rad}{s}$

Those bounds could of course be chosen differently, which always depends on the specifications of the application. In this case they are chosen as an example to demonstrate the problems and challenges of MPC stability theory.

4.9.1 MPC with Unconstrained Cost Function

In practise, the main issue of MPC with Unconstrained Cost Function is that it requires the Control and Prediction Horizon to have the same length. As the Control Horizon is much more computationally demanding than the Prediction Horizon, this could limit the feasible set dramatically. To compensate for that the terminal constraint set should be as large as possible. Another practical issue is how to determine the terminal constraint set. In the case of the MPC with the Unconstrained Cost Function, this is relatively simple: When the constraints are not active, this MPC becomes equivalent to a Linear Quadratic Regulator (see 4.1.4.3). Consequently, it is sufficient to determine the set with non-active constraints to be used as terminal constraint set. Due to the properties of the LQR all properties of the terminal constraint set are then automatically fulfilled in this set.

The downside of this strategy is that if the terminal constraint set is very large, the constraints are not active most of the time. Consequently, the controller is equivalent to a Linear Quadratic Regulator and the advantage of MPC, which is the constraint handling, is lost. It also means that the controller might become very slow in the attempt of increasing the set with nonactive constraints, which mainly means to slow the controller down by increasing the input weight R. In that case it might be more feasible to implement a LQR in the first place, which is much easier to handle.

In order to determine the terminal constraint set, a functionality of the Multi-Parametric Toolbox is used. For further details about this toolbox refer to Appendix A.2. Unfortunately, the build in function mpt_invariantSet cannot be used directly, as it is designed for explicit MPC. Therefore, a workaround has been implemented using subfunctions of mpt_invariantSet.

To calculate the backwards reachable set, Algorithms 1 and 2 are used, which are described above.

4.9.1.1 Pitch Axis

In order to implement an MPC with Unconstrained Cost Function, the pitch controller used before has to be modified. First of all, a terminal weight $Q_{pitch,terminal}$ equal to the solution of the Riccatti equation has to be added. Secondly, the Control and Prediction horizon have to have the same length now, thus $H_c = H_p = 4$:

Parameter	Value
Q_{pitch}	$\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$
$Q_{pitch,terminal}$	$\begin{bmatrix} 182.9 & 3.6 \\ 3.6 & 2.6 \end{bmatrix}$
R_{pitch}	0.01
$\triangle R_{pitch}$	$1 \cdot 10^{-5}$

This modified controller still shows very similar performance to the original pitch controller.

For the stability analysis, the first step now is to determine the terminal constraint set. The set determined by the Multi-Parametric Toolbox is shown in Figure 4.6.

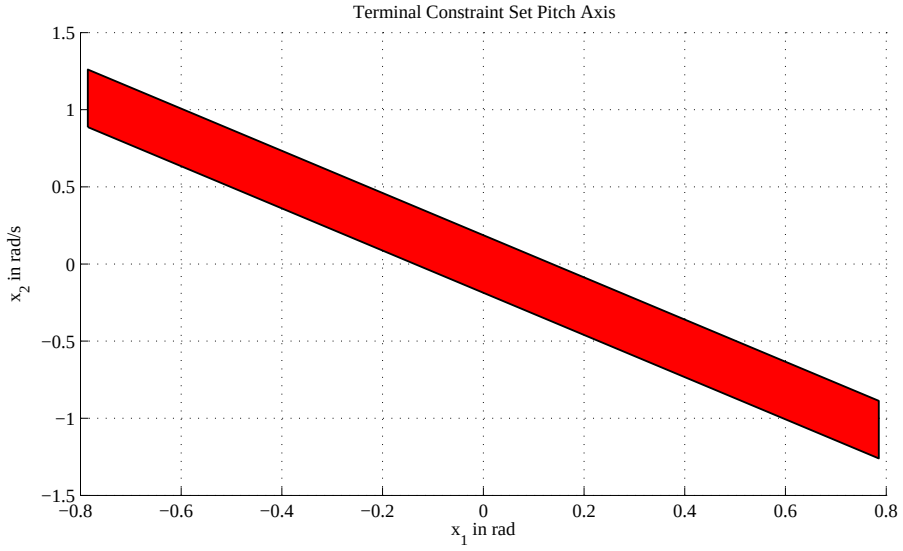

Figure 4.6: Terminal Constraint Set Pitch Axis, LQR.

Now in the second step, the backwards reachable set of this terminal set is calculated. The result is shown in Figure 4.7.

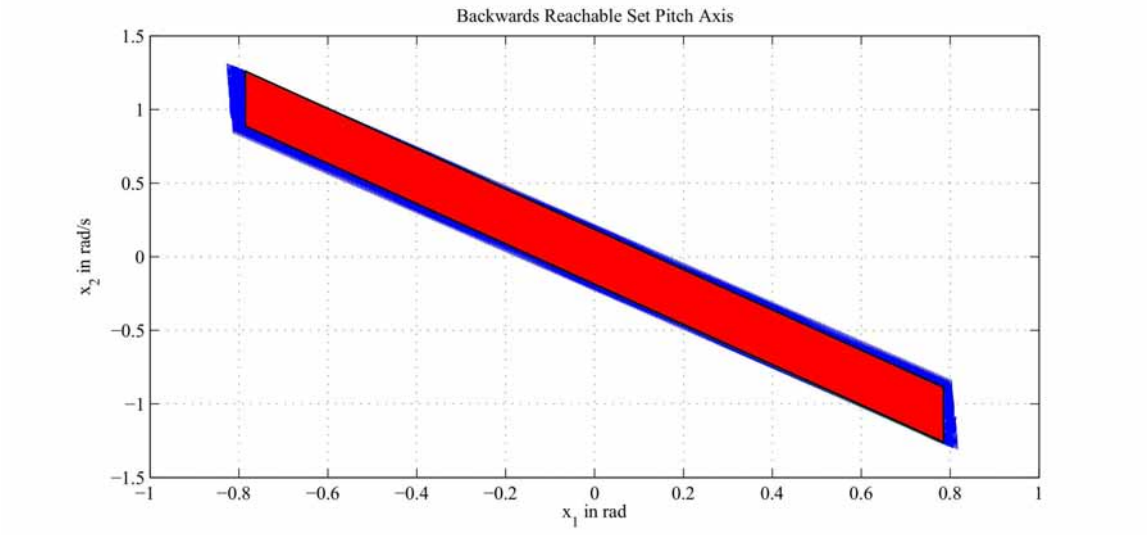

Figure 4.7: Backwards Reachable Set Pitch Axis, LQR.

Figure 4.7 shows that the specifications of $\Delta x_1 = \pm 0.5 rad$ and $\Delta x_2 = \pm 0.25 rad$ are not met by this terminal set. It is also important to note that the subset where the constraints are active (blue set) is very small.

To meet the specification, the horizon should be much larger. Unfortunately this is computationally not possible. The second possibility is to slow down the controller to increase the terminal set to meet the specifications. Increasing the Input Weight to $R = 0.2$, the following terminal set can be calculated:

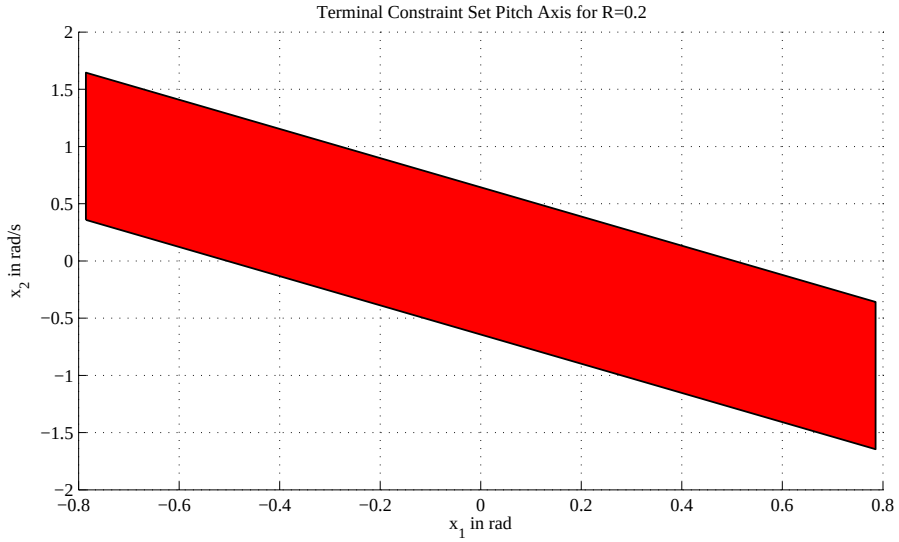

Figure 4.8: Terminal Constraint Set Pitch Axis R=0.2, LQR.

This set already meets the specifications of $\Delta x_1 = \pm 0.5 rad$ and $\Delta x_2 = \pm 0.25 rad$. So nominal stability can actually be shown for this modified controller. Unfortunately, this controller turned out to be very slow. Additionally, as the terminal set is so large, this MPC is equivalent to a Linear Quadratic Regulator all the time. So in this case it is unnecessary to use an MPC and an LQR could be implemented instead.

4.9.1.2 Elevation Axis

As for the pitch axis, a terminal weight has to be added to the original elevation controller and the horizons set to $H_c = H_p = 5$. Thus

Parameter	Value
Q_{elev}	$\begin{bmatrix} 4 \cdot 10^4 & 0 \\ 0 & 4 \cdot 10^3 \end{bmatrix}$
$Q_{elev,terminal}$	$10^6 \cdot \begin{bmatrix} 6.36 & 0.0066 \\ 0.0066 & 0.0041 \end{bmatrix}$
R_{elev}	$1 \cdot 10^{-4}$
$\triangle R_{elev}$	$1 \cdot 10^{-5}$

For the stability analysis, the first step now is to determine the terminal constraint set. Using the Multi-Parametric Toolbox the terminal constraint set shown in Figure 4.9 is derived.

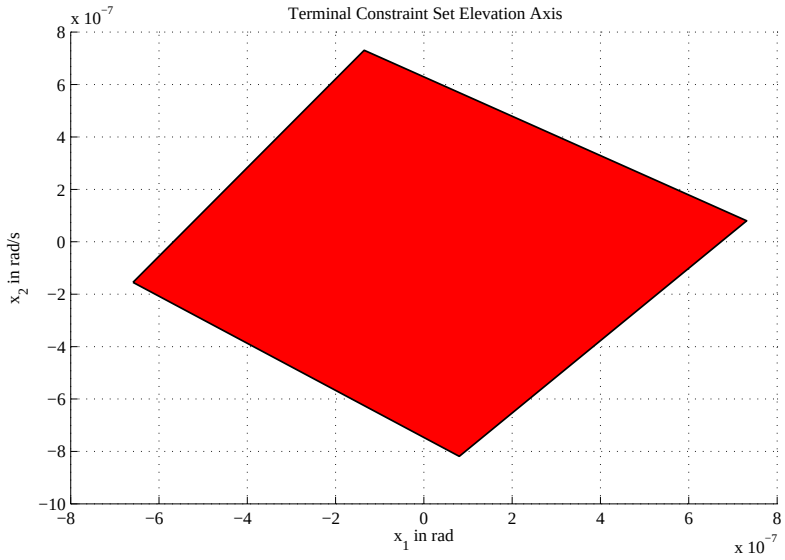

Figure 4.9: Terminal Constraint Set Elevation Axis, LQR.

The terminal constraint set shown in Figure 4.9 is very small and can therefore be neglected. Instead, only the origin is used as a terminal constraint set. This results in the backwards reachable set shown in Figure 4.10.

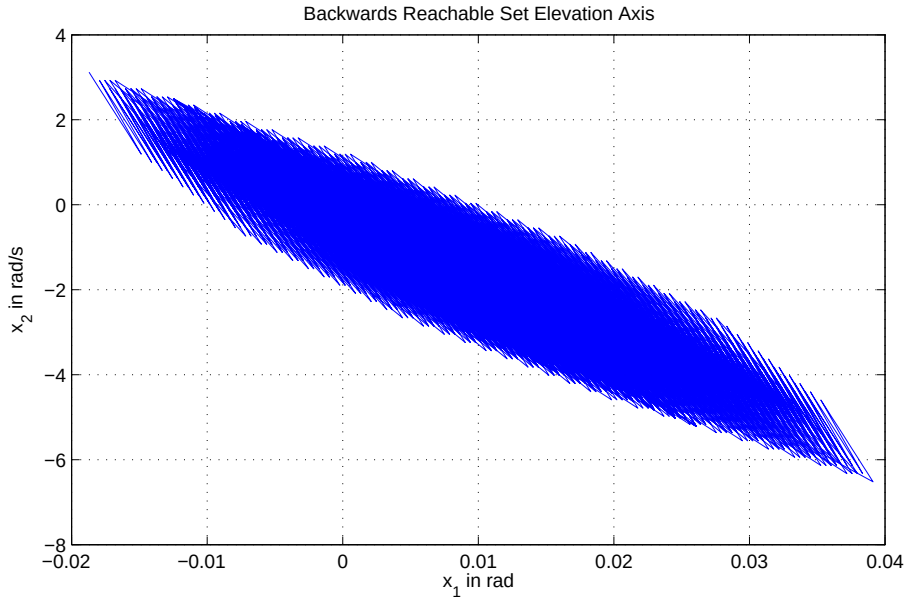

Figure 4.10: Backwards Reachable Set Elevation Axis, LQR.

This terminal set does not meet the minimal requirements for the terminal constraint set either. As for the pitch controller, longer horizons would be needed to fulfill the specifications.

The input weight R could now be increased to get a larger terminal constraint set in order to meet the specifications. Unfortunately, the Multi-Parametric Toolbox did not return a larger set for any choice of R. The reason seems to be that the Toolbox cannot handle asymmetric constraint sets. The state constraints on the pitch axis were symmetric to the origin with $-\frac{\pi}{4}rad \leq x_1 \leq \frac{\pi}{4}rad$, whereas the constraints on the elevation axis are not with $-0.7154rad \leq x_3 \leq 0.2rad$.

4.9.1.3 Travel Axis

As before, a terminal weight has to be added to the original travel controller and the horizons are set to $H_c = H_p = 4$. Thus

Parameter	Value
Q_{travel}	$\begin{bmatrix} 0.04 & 0 \\ 0 & 0.015 \end{bmatrix}$
$Q_{travel,terminal}$	$\begin{bmatrix} 3.45 & 0.24 \\ 0.24 & 0.17 \end{bmatrix}$
R_{travel}	0.00001
$\triangle R_{travel}$	$1 \cdot 10^{-5}$

Unfortunately, as for the elevation axis, the Multi-Parametric Toolbox only determines an empty set as terminal constraint set. Thus the origin has to be used as terminal constraint set again. This results in the backwards reachable set shown Figure 4.11.

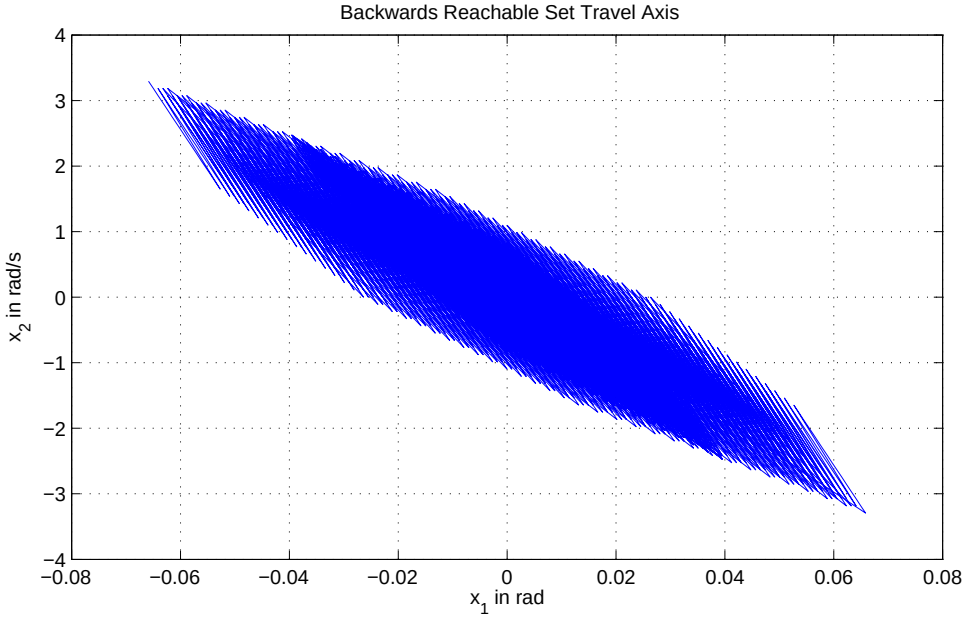

Figure 4.11: Backwards Reachable Set Travel Axis, LQR.

This terminal set does not meet the requirement of $\Delta x_5 \pm 0.5rad$. To get a larger terminal constraint set the speed of the MPC is decreased by increasing the input weight from $R = 0.0001$ to $R = 0.1$. This results in the following terminal constraint set:

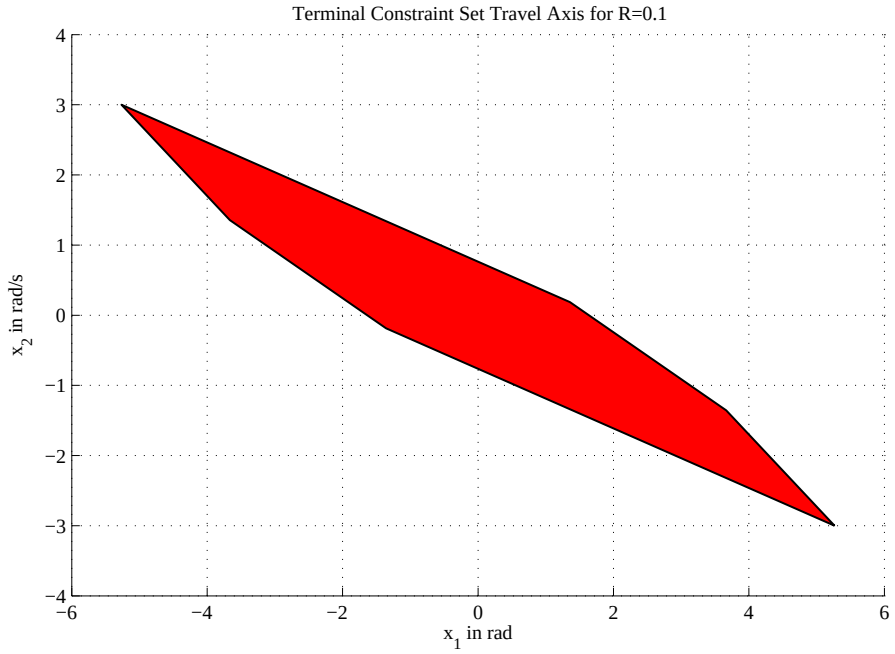

Figure 4.12: Terminal Constraint Set Travel Axis for $R = 0.1$.

As for the pitch axis, this terminal set already meets the requirements. Unfortunately, this controller turned out to be extremely slow and is again equivalent to a Linear Quadratic Regulator.

The implementation of the MPC with an Unconstrained Cost Function has shown that the requirement that control and prediction horizon have to have the same length is indeed problematic as it limits the length of the horizons. Either the feasible set does not meet the requirements or the controller is very slow and equivalent to an LQR. The proceeding methods try to counteract these problems, because they enable control and prediction horizon to have differing lengths. Consequently, a long prediction horizon can be chosen to achieve a large enough feasible set.

4.9.2 Dual Mode MPC

The idea of Dual Mode MPC is to switch from the MPC to an alternative controller when entering the terminal constraint set. This alternative controller is chosen such that the

requirements of a terminal constraint set are met. In theory, there are many possibilities for this alternative controller. In practise though there are additional requirements. The transition from the MPC to the alternative controller has to be sufficiently smooth. A reasonable approach is to switch to the alternative controller when the MPC is entering a set where the constraints are not active any more. Thus the MPC will only be active for active constraints, which is a good property as the MPC has its strength in the constraint handling. Another reason to use this strategy is that the Multi-Parametric Toolbox can only handle an explicit closed-loop representation of the MPC to calculate the terminal constraint set. This closed-loop representation can only be determined for the unconstrained MPC though (unconstrained MPC here means a state space representation of the MPC in the case of nonactive constraints). The unconstrained MPC can be derived using the Matlab MPC Toolbox and the routine ss. Using this state space representation, the closed-loop system can be derived. Then the Multi-Parametric Toolbox is used to determine the terminal constraint set.

With this strategy the original Model Predictive Controllers for each axis do not have to be modified either.

4.9.2.1 Pitch Axis

For the Pitch Controller the following state space representation of the unconstrained MPC can be derived using the routine Matlab routine ss:

$$
\begin{aligned}
z_{k+1} = &\begin{bmatrix} 0.9862 & 0.0008572 & 0 & -0.007143 & 0 \\ -0.9936 & 0.002553 & 0 & -0.6886 & 0 \\ -0.001081 & -0.007694 & 1 & -0.007694 & 0 \\ 0.007871 & -0.001111 & 0 & 0.9989 & 0 \\ -19.64 & -18.91 & -1 & -12.71 & 0 \end{bmatrix} z_k \\
&+ \begin{bmatrix} 0.01376 & 0.007143 \\ -0.00826 & -0.007777 \\ 0.001081 & 0.007694 \\ -0.007871 & 0.001111 \\ -0.4592 & -1.259 \end{bmatrix} u_k \\
y_k = &\begin{bmatrix} -19.64 & -18.91 & -1 & -12.71 & 0 \end{bmatrix} z_k + \begin{bmatrix} -0.4592 & -1.259 \end{bmatrix} u_k
\end{aligned}
$$

In the next step the closed-loop system can be calculate with the routine feedback(sys1,sys2). This results in the following state space model:

$$
\begin{aligned}
z_{k+1} = &\begin{bmatrix}
1 & 0.008 & 0 & 0 & 0 & 0 & 0 \\
0.02289 & 1.063 & 0.979 & 0.9425 & 0.04985 & 0.6336 & 0 \\
0.01376 & 0.007143 & 0.9862 & 0.0008572 & 0 & -0.007143 & 0 \\
-0.00826 & -0.007777 & -0.9936 & 0.002553 & 0 & -0.6886 & 0 \\
0.001081 & 0.007694 & -0.001081 & -0.007694 & 1 & -0.007694 & 0 \\
-0.007871 & 0.001111 & 0.007871 & -0.001111 & 0 & 0.9989 & 0 \\
-0.4592 & -1.259 & -19.64 & -18.91 & -1 & -12.71 & 0
\end{bmatrix} z_k \\
&+ \begin{bmatrix}
0.01376 & 0.007143 \\
-0.00826 & -0.007777 \\
0.001081 & 0.007694 \\
-0.007871 & 0.001111 \\
-0.4592 & -1.259
\end{bmatrix} u_k \\
y_k = &\begin{bmatrix} -19.64 & -18.91 & -1 & -12.71 & 0 \end{bmatrix} z_k + \begin{bmatrix} -0.4592 & -1.259 \end{bmatrix} u_k
\end{aligned}
$$

This representation makes it possible to calculate the terminal constraint set using the Multi-Parametric Toolbox. Unfortunately, the Toolbox only returns an empty set as terminal set. This makes the Dual Mode MPC on the pitch axis equivalent to the Terminal State MPC investigated in the next part.

4.9.2.2 Elevation

For the Elevation Controller the following state space representation of the unconstrained MPC can be derived using the routine ss:

$$z_{k+1} = \begin{bmatrix} 0.996 & 0.001714 & -0.004023 & -0.0002401 & 0 \\ -2.806 & -0.0004213 & -2.806 & -0.9543 & 0 \\ -0.001748 & -0.0009653 & 0.9983 & -0.0009653 & 0 \\ 0.0009663 & -0.001747 & 0.0009663 & 0.9983 & 0 \\ -2472 & -880.3 & -2474 & -839.8 & 0 \end{bmatrix} z_k + \begin{bmatrix} 0.001209 & -0.0007166 \\ -0.007415 & -0.002378 \\ 0.001748 & 0.0009653 \\ 0.0009663 & 0.001747 \\ -7.611 & -4.166 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} -2472 & -880.3 & -2474 & -839.8 & 0 \end{bmatrix} z_k + \begin{bmatrix} -7.611 & -4.166 \end{bmatrix} u_k$$

In the next step the closed-loop system can be calculate with the routine feedback(sys1,sys2). This results in the following state space model:

$$z_{k+1} = \begin{bmatrix} 0.996 & 0.00171 & -0.00402 & -0.00024 & 0 & -0.00120 & 0.000716 \\ -2.806 & -0.000421 & -2.8 & -0.954 & 0 & 0.00741 & 0.00237 \\ -0.00174 & -0.000965 & 0.998 & -0.000965 & 0 & -0.00174 & -0.000965 \\ 0.000966 & -0.00174 & 0.000966 & 0.998 & 0 & 0.000966 & -0.00174 \\ -2472 & -880.3 & -2474 & -839.8 & 0 & 7.61 & 4.166 \\ -0.00280 & -0.000997 & -0.00280 & -0.00095 & 0 & 1 & 0.002 \\ -2.803 & -0.997 & -2.805 & -0.952 & 0 & 0.00640 & 1.005 \end{bmatrix} z_k + \begin{bmatrix} 0.001209 & -0.0007166 \\ -0.007415 & -0.002378 \\ 0.001748 & 0.0009653 \\ -0.0009663 & 0.001747 \\ -7.611 & -4.166 \\ 0 & 0 \\ 0.008627 & -0.004722 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} -2472 & -880.3 & -2474 & -839.8 & 0 & 7.61 & 4.16 \end{bmatrix} z_k + \begin{bmatrix} -7.61 & -4.16 \end{bmatrix} u_k$$

Unfortunately, as for the pitch axis, the Toolbox only returns an empty set as terminal set. So again, the Dual Mode MPC on the elevation axis equivalent to the Terminal

State MPC investigated in the next part.

4.9.2.3 Travel

For the Travel Controller the following state space representation of the unconstrained MPC can be derived using the routine ss:

$$z_{k+1} = \begin{bmatrix} 0.9874 & 0.003629 & 0 & -0.00437 & 0 \\ -0.2985 & 0.002979 & 0 & -0.9594 & 0 \\ -0.004178 & -0.006735 & 1 & -0.006735 & 0 \\ 0.006764 & -0.0042 & 0 & 0.9958 & 0 \\ -34.29 & -117.5 & -0.9542 & -113.1 & 0 \end{bmatrix} z_k + \begin{bmatrix} 0.01261 & 0.00437 \\ 0.002795 & -0.005344 \\ 0.004178 & 0.006735 \\ -0.006764 & 0.0042 \\ -0.9874 & -2.002 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} -34.29 & -117.5 & -0.9542 & -113.1 & 0 \end{bmatrix} z_k + \begin{bmatrix} -0.9874 & -2.002 \end{bmatrix} u_k$$

In the next step the closed-loop system can be calculate with the routine feedback(sys1,sys2). This results in the following state space model:

$$z_{k+1} = \begin{bmatrix} 0.987 & 0.00362 & 0 & -0.00437 & 0 & -0.0126 & -0.00437 \\ -0.298 & 0.00297 & 0 & -0.959 & 0 & -0.00279 & 0.00534 \\ -0.00417 & -0.00673 & 1 & -0.00673 & 0 & -0.00417 & -0.00673 \\ 0.00676 & -0.0042 & 0 & 0.995 & 0 & 0.00676 & -0.0042 \\ -34.29 & -117.5 & -0.954 & -113.1 & 0 & 0.987 & 2.002 \\ -0.00115 & -0.00394 & -3.2e-05 & -0.00379 & 0 & 1 & 0.00806 \\ -0.2874 & -0.985 & -0.00799 & -0.947 & 0 & 0.00827 & 1.017 \end{bmatrix} z_k + \begin{bmatrix} 0.01261 & 0.00437 \\ 0.002795 & -0.005344 \\ 0.004178 & 0.006735 \\ -0.006764 & 0.0042 \\ -0.9874 & -2.002 \\ -3.311e-05 & -6.713e-05 \\ -0.008278 & -0.01678 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} -34.29 & -117.5 & -0.954 & -113.1 & 0 & 0.987 & 2 \end{bmatrix} z_k + \begin{bmatrix} -0.987 & -2 \end{bmatrix} u_k$$

This representation makes it possible to calculate the terminal constraint set using the Multi-Parametric Toolbox. As for pitch and elevation axis the Multi-Parametric Toolbox returns an empty set as terminal set as well.

So summing up, the implementation of the Dual Mode MPC was not successful. The problem is that the state space models of the closed-loop systems are very complex. As a result, the Multi-Parametric Toolbox has problems finding the set with inactive constraints. So it returns an empty set. It is to be assumed though that the set with inactive constraints is actually larger than only the origin. As no other algorithm than the Mutli-Parametric Toolbox was available to calculate this set in this thesis, there is no other possibility than to use the origin as a terminal constraint set. In that case though, the Dual Mode MPC becomes equivalent to the Terminal State Method described next.

4.9.3 Terminal State

In the terminal state method the origin is chosen as the terminal constraint set. The advantage of this method is that it is very simple to implement. Additionally, the prediction and control horizon do not have to have the same length. Compared to the Dual Mode MPC the problem of finding a terminal constraint set is also obsolete. The disadvantage of this method is that the feasible set is smaller as it is for the Dual Mode Controller, as the terminal constraint set is very small.

4.9.3.1 Pitch Axis

In order to calculate the feasible set, the backwards reachable set was calculated using Definition 1. Because the prediction horizon $H_p = 80$ is not equal to the control horizon $H_c = 4$, the backwards reachable set has to be calculated in two steps: Firstly, the last $H_p - H_c = 76$ steps are calculated, where the control input is constant. Using the function in Appendix 4.1.4.4, this yields the following set:

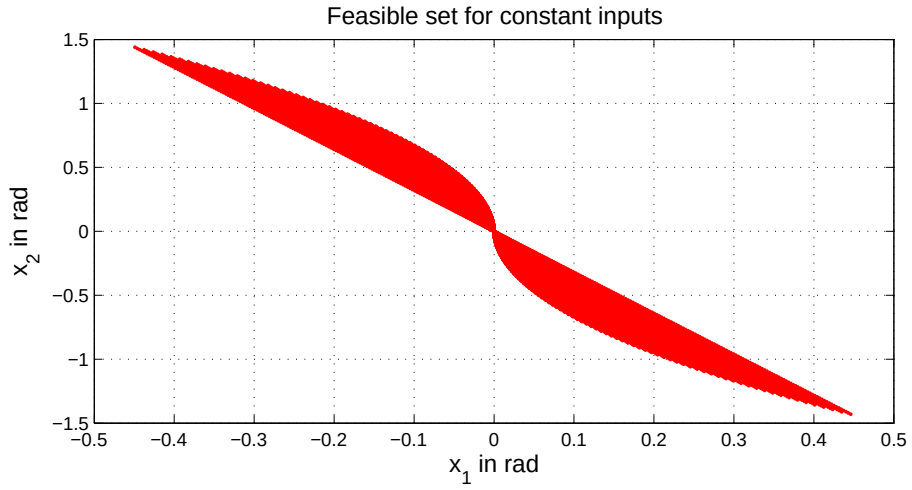

Figure 4.13: Feasible Set Pitch Axis for a Constant Input.

If now the last 4 steps are added to the set, this yields:

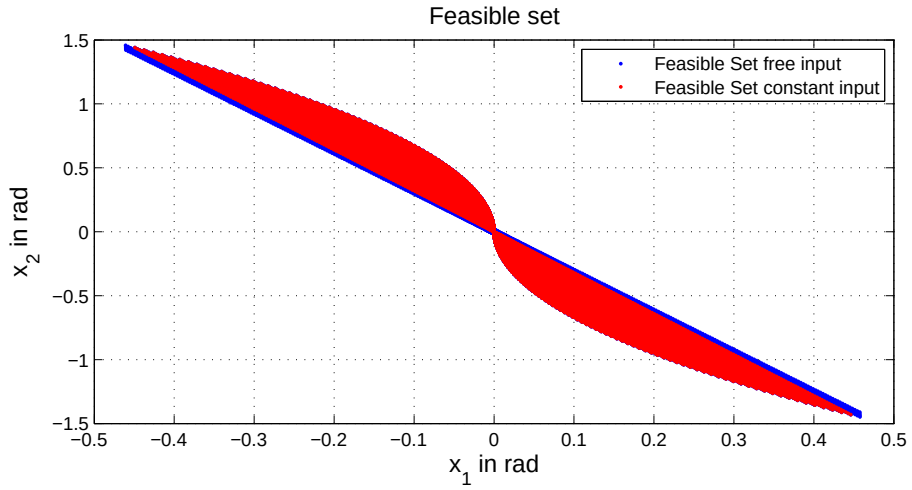

Figure 4.14: Feasible Set Pitch Axis.

Figure 4.14 shows that the requirements of $\Delta x_1 = \pm 0.5 rad$ and $\Delta x_2 = \pm 0.25 rad$ are not met. The problem is that most of the time the feasible set has to be symmetric to the origin and convex. The smallest convex subset around the origin in Figure 4.14 is very

small. Consequently, the longer prediction horizons used in the terminal state method are not helping to actually increase the feasible set in this case.

4.9.3.2 Elevation Axis

In order to calculate the feasible set, the backwards reachable set was calculated using Definition 1. Because the prediction horizon $H_p = 15$ is not equal to the control horizon $H_c = 5$, the backwards reachable set has to be calculated in two steps: Firstly, the last $H_p - H_c = 10$ steps are calculated, where the control input is constant. Using the function in Appendix 4.1.4.4, this yields the following set:

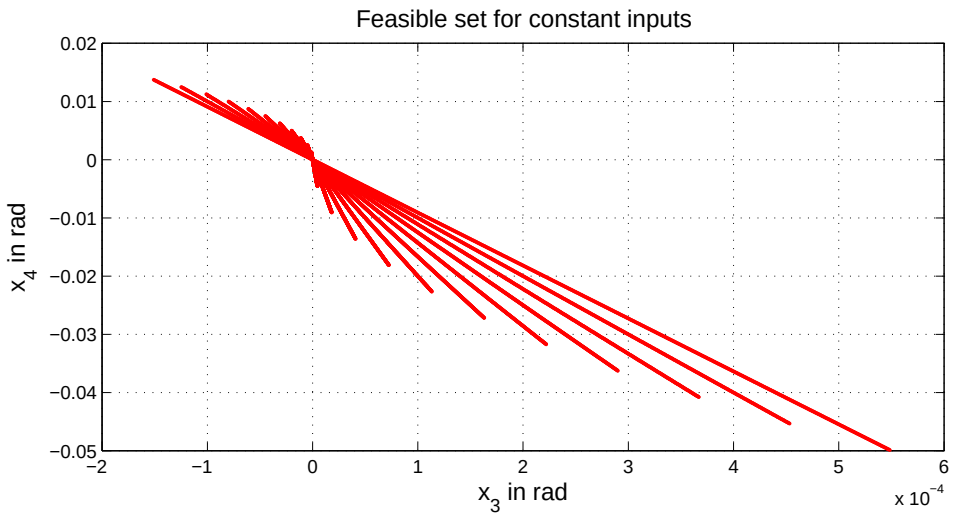

Figure 4.15: Feasible Set Elevation Axis for a Constant Input.

If the last 4 steps are added to the set now, this yields:

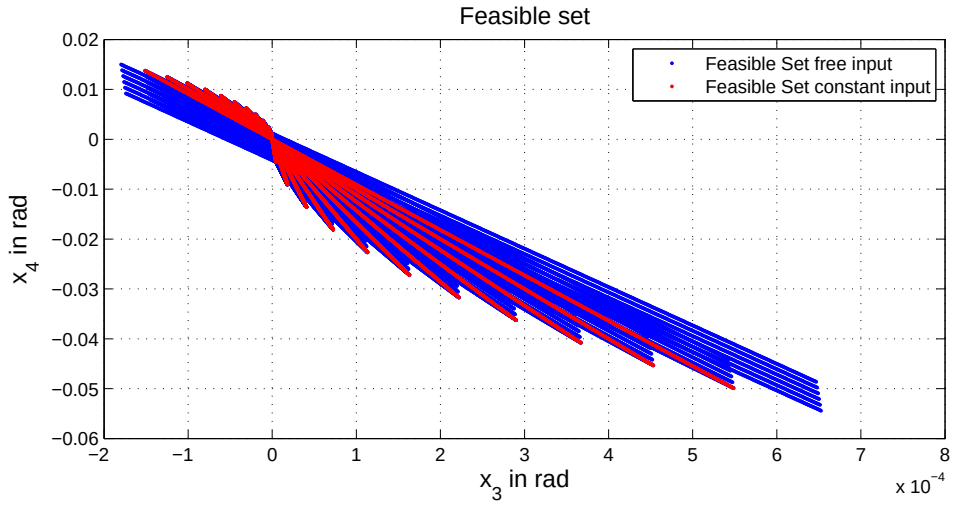

Figure 4.16: Feasible Set Elevation Axis.

As before, the feasible set depicted in Figure 4.16 is not convex. The smallest convex subset around the origin is rather small. As a result, the requirement of $\Delta x_3 = \pm 0.5 rad$ and $\Delta x_4 = \pm 0.25 rad$ are not met at the same time.

4.9.3.3 Travel Axis

In order to calculate the feasible set, the backwards reachable set was calculated using Definition 1. Because the prediction horizon $H_p = 80$ is not equal to the control horizon $H_c = 4$, the backwards reachable set has to be calculated in two steps: Firstly, the last $H_p - H_c = 76$ steps are calculated, where the control input is constant. Using the function in Appendix 4.1.4.4, this yields the following set:

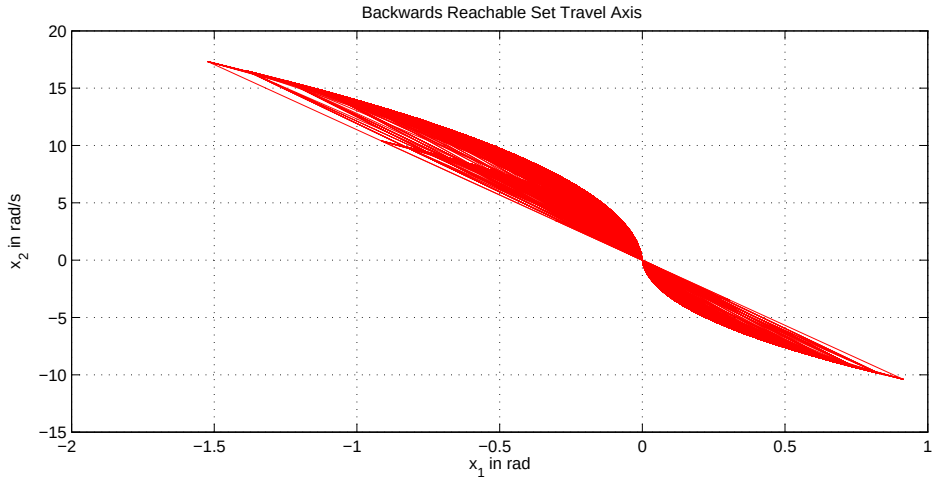

Figure 4.17: Feasible Set Travel Axis for a Constant Input.

If now the last 4 steps are added to the set, this yields:

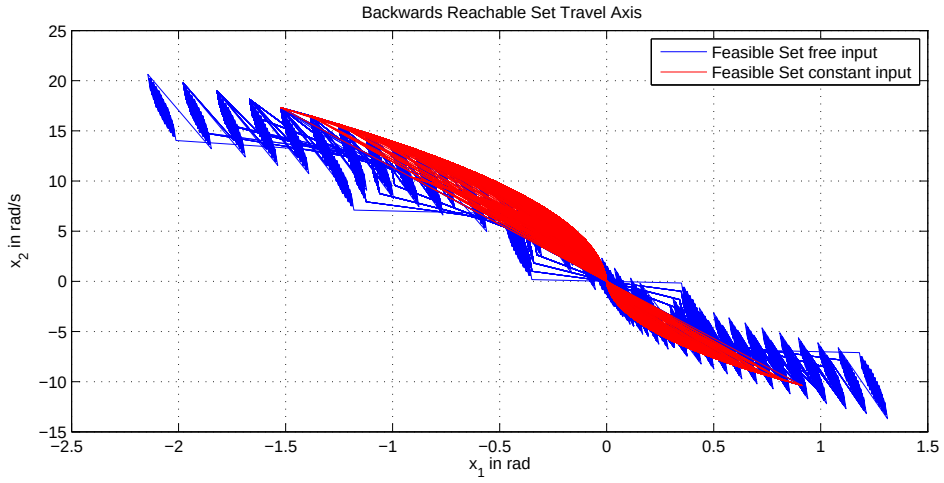

Figure 4.18: Feasible Set Elevation Axis.

As before, the feasible set depicted in Figure 4.16 is not convex. The smallest convex subset around the origin is rather small. As a result, the requirement of $\Delta x_5 = \pm 0.5 rad$ and $\Delta x_6 = \pm 0.25 rad$ are not met at the same time.

All in all, the advantage of using longer prediction horizons in the terminal state method does not necessarily results in a larger feasible set. The backwards reachable set is indeed much larger than for the other methods, unfortunately it is not convex and only contains a very small convex subset. The reason for this non convexity lays in the fact that beyond the control horizon, the control input has to stay the same. So let's consider the pitch axis of the helicopter. The pitch angle might be deviated by a certain angle from the reference trajectory without any angular speed. To steer it back, the helicopter has to be first accelerated around the pitch axis and then slowed down again to stop at the reference trajectory and not shoot over it. This slowing down is the problem when the control input has to stay the same beyond the control horizon: Then, it is not possible to first accelerate and then slow down again, because different input signals are needed for that. The results might be better for a system with a lot of damping, where the input signal does not necessarily has to be changing to stop at the reference trajectory.

4.9.4 Summary

All three methods investigated to prove nominal stability turned out to be problematic when applied to the Quanser helicopter.

The MPC with an Unconstrained Cost suffered from the fact that the terminal constraint set was either too small, could not be calculated at all or it had the right size, but resulted in an impractically slow controller. The most important issue is that the control and prediction horizon have to have the same length in this method and therefore are very small due to computational limitations. In consequence, the backwards reachable set becomes very small. So even if the terminal constraint set has is large enough, the feasible set will not be much larger than this set. In consequence, the MPC is equivalent to an LQR most of the time.

The Dual Mode MPC caused problems in the implementation. The state space representation of the unconstrained MPC was too complex for the Multi-Parametric Toolbox to calculate a terminal constraint set larger than the origin. So in principle, this method might give good results, but could not be implemented in this application with the available tools.

The Terminal Set Method has the advantage that control and prediction horizon can be chosen differing. Thus much longer prediction horizons could be used for this method. This resulted in comparably large backwards reachable sets. Unfortunately, those sets very highly non-convex. The largest convex subsets of those sets were very small as well. This is due to the fact that beyond the control horizon, the input signal cannot be changed any more. So the strategy of using small control horizon and large prediction horizons might actually not result in larger feasible sets in most applications.

5 Evaluation

This chapter evaluates the properties and the performance of the designed model predictive controller. To illustrate the properties of the cascaded MPC as a whole, the individual model predictive controllers are examined. Step response experiments are used to illustrate the closed-loop behaviour of each axes and key characteristics such as the time constant are derived. Additionally, the benefit of the input disturbance estimator in achieving offset-free control and increasing performance is shown.

A benchmark scenario is used to evaluate the overall performance of the controller. The scenario was introduced by the Institute for System Theory and Automatic Control at University of Stuttgart in control laboratories. It was also used as a benchmark scenario in thesis projects at University of Stuttgart before.

5.1 Controller Performance

Step response simulations and experiments can be used to determine several characteristics of the controller's performance. Those are:

- Time Constant
- Overshoot
- Settling time
- Control offset

The benefit of the input disturbance estimator concerning control offset will also be shown.

Because the controller has been divided into individual controllers for every axis, separate step response simulation and experiments will be conducted for every individual axis.

5.1.1 Controller Characteristics

Time Constant The so called initial slope method is used to determine the time constant. If a tangent is drawn on the initial slope of the step response, the intersection of this tangent and the final value of the step approximates the time constant.

Overshoot The overshoot is defined as the maximum overshoot over the final value of the step response

Settling Time The following definition of the settling time will be used: The settling time is the time until the controlled variable stays within 10% of the reference target.

5.1.1.1 Pitch Controller

The following reference step was applied to the pitch axis: The initial pitch angle is $\theta_0 = -\frac{\pi}{4}$. At $t = 5s$ the reference angle jumps to $\theta_{final} = \frac{\pi}{4}$. Figure 5.1 shows the results of the simulation and experiment. The first graph shows the reference angle and the measured pitch angles in the simulation and experiment respectively. The behaviour in simulation and experiment is quite similar. It is notable though that the experimental results show a slight oscillation. The applied input signals in simulation and experiment (second and third plot in Figure 5.1) also reflect the oscillation in the experiment. Both in simulation and experiment, the input signal first jumps to the maximum value of $1.6N$. In the experiment though, the controller holds that value significantly longer and

then shows a declining oscillation. This discrepancy between simulation and experiments results from the fact that the motors have a certain delay from the impressed voltage to the generated thrust. This had not been modeled. For the controller this delay results in the fact that the initial value of $1.6N$ is hold longer, because the system does not react immediately: If you look at the measured pitch angle in Figure 5.1, you can see that in the experiment, the system does not react for about the first $0.1s$. As a result, the controller holds the initial value for the input signal longer. The proceeding oscillation also occurs because of this delay, as the thrust does not follow the impressed voltage right away. This is a common problem in systems with delay.

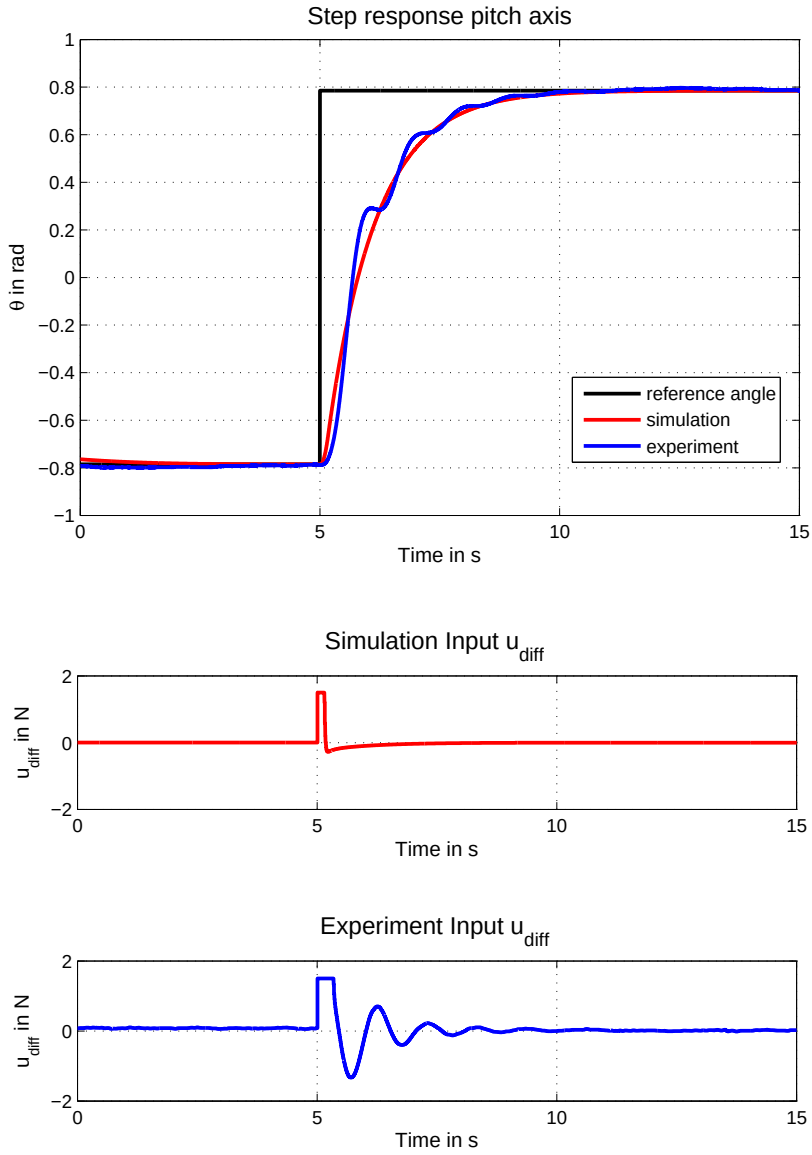

Figure 5.1: Response to a Step in the Reference Signal from $-\frac{\pi}{4}$ to $\frac{\pi}{4}$ in Simulation and Experiment.

Time Constant Using the initial slope method in Figure 5.2, the time constant can be approximated as $\tau_{pitch} \approx 1.2s$. Only the simulation result was used in Figure 5.2, as they are almost identical to the experimental results.

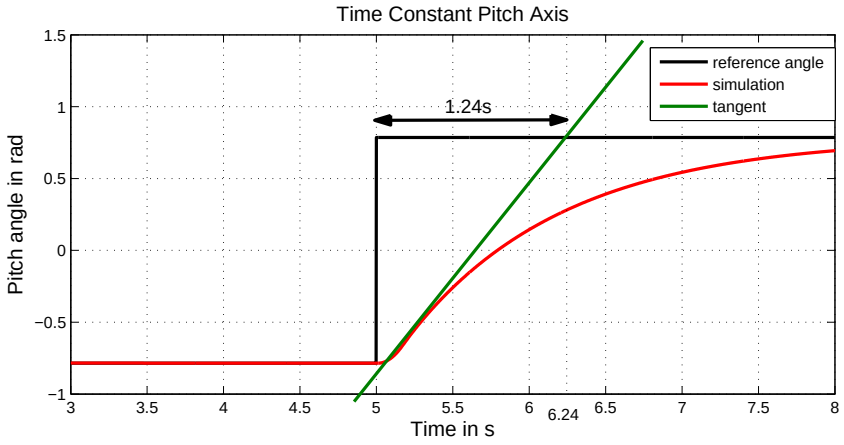

Figure 5.2: Time Constant Pitch Axis Using Initial Slope Method.

Overshoot Figure 5.1 shows that there is no overshoot on the pitch axis.

Settling time Settling time can be define by the time it takes for the controller to reach $\epsilon = 10\%$ of the final value. With a step of $\Delta\theta = \frac{\pi}{2}$ this means $\epsilon = 10\% \cdot \frac{\pi}{2}$rad=0.1571rad. Figure 5.3 shows that the settling time is about $t_{settle} = 2.4s$.

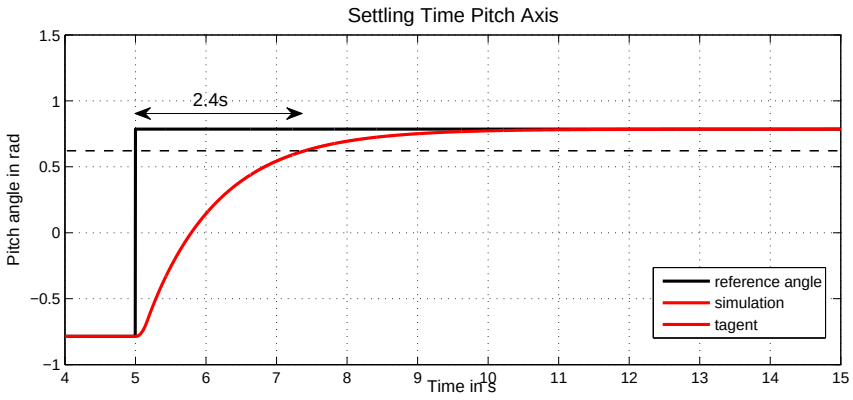

Figure 5.3: Settling Time Pitch Axis.

5.1.1.2 Elevation Controller

To determine the characteristics of the elevation axis controller, a step response experiment was conducted: The elevation angle starts at an initial value of $\phi_0 = -0.5$. At $t = 5s$ the reference angle switches to $\phi_{step} = 0.2$. In Figure 5.4 the results in the simulation and experiment are depicted.

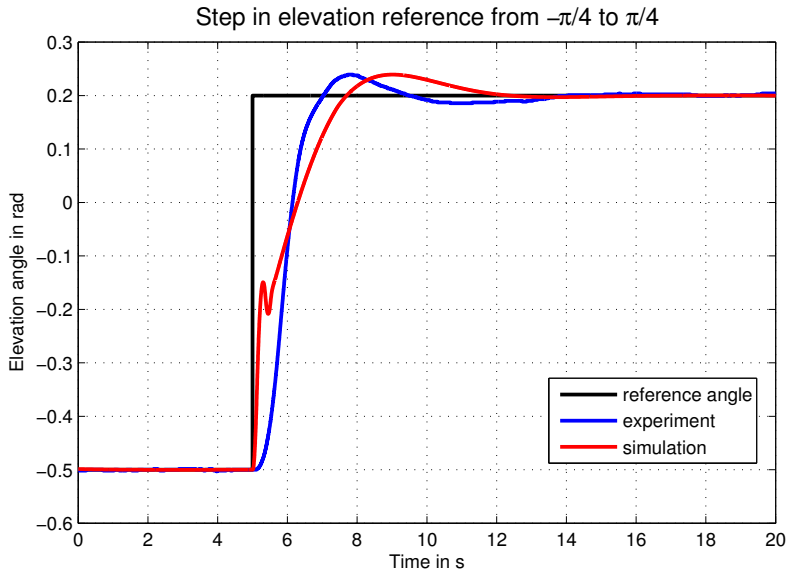

Figure 5.4: Step on Elevation Angle from $-0.5rad$ to $0.2rad$.

Time Constant As for the pitch axes, the initial slope method is used to determine the time constant of the closed-loop system. Figure 5.5 shows that the time constant is $\tau_{elev} \approx 1.3s$.

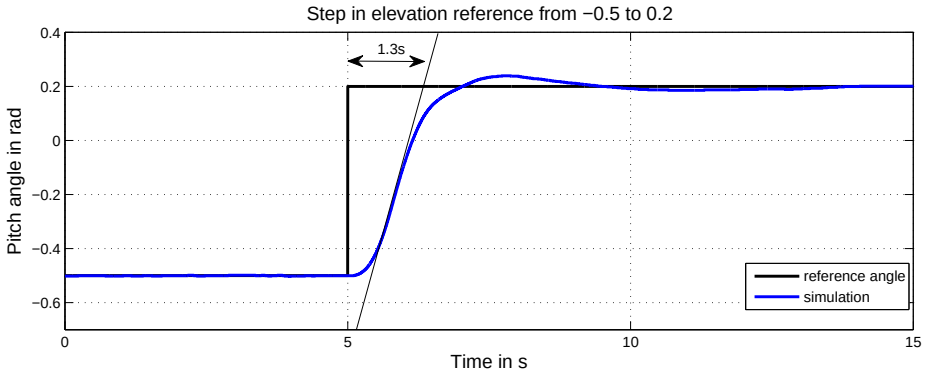

Figure 5.5: Time Constant Elevation Axis.

Overshoot The overshoot can be measured as $\Delta\phi = 0.04rad$ using Figure 5.6.

Settling time Settling time can be define by the time it takes for the controller to reach $\epsilon = 10\%$ of the final value. With a step of $\Delta\phi = 0.7$ this means $\epsilon = 0.07$. Figure 5.6 shows that the settling time is $t_{settle} = 2.4s$.

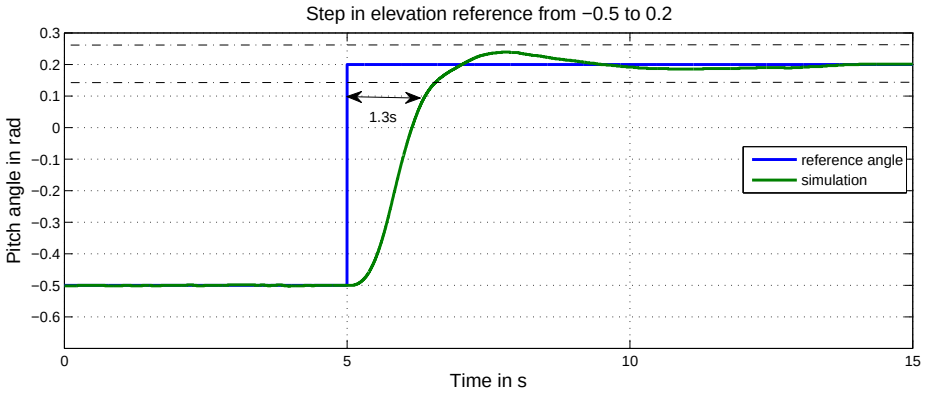

Figure 5.6: Settling Time Elevation Axis.

5.1.2 Offset Free Control

To achieve offset free control, a disturbance estimator has been used. The estimated disturbance was added to the MPC as a measured disturbance. This section will investigate the benefit of the disturbance estimator in achieving offset free control. Control scenarios with input disturbances are used to cause control offsets. The disturbance estimator is then used to counteract those control offsets.

5.1.2.1 Pitch axis

Errors in the relation between the impressed voltage and the aerodynamic force, aerodynamic disturbances, wind resistance, ground effect and other disturbances result in a control offset in the MPC.

Let's consider the following scenario: The MPC is to control the pitch angle to zero. At $t_{dist} = 5s$ an input disturbance of $\Delta u_{diff} = 1N$ is added to the system. Figure 5.7 shows that this input disturbance results in a significant control offset of $\Delta\theta = 1rad$ on the pitch angle. Additionally, the experimental results also show a control offset for

the first five seconds. This is due to actual input disturbances in the plant, most likely because the measured thrust of each motor does not perfectly represent the actual thrust generated by the individual motors.

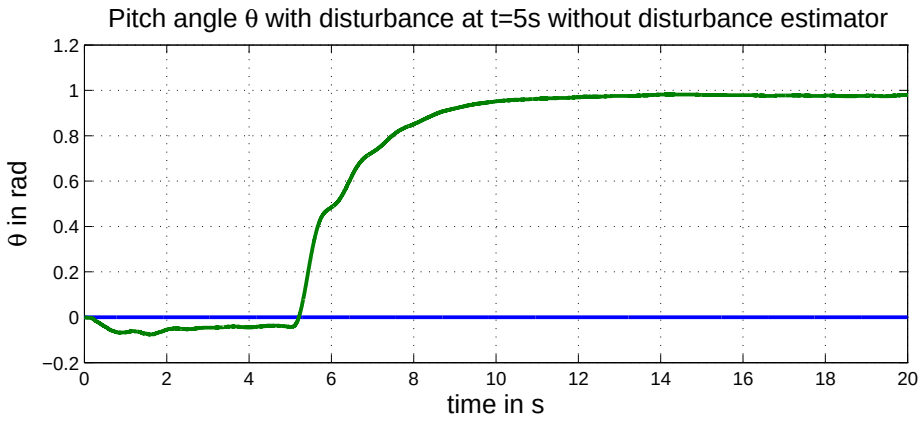

Figure 5.7: Pitch Angle with Disturbance at t=5s without Estimator.

In an other experiment, the disturbance estimator is added to the controller. Figure 5.8 shows that the added input disturbance still results in an initial control offset of about $0.5rad$. But when the disturbance estimator is reaching the exact estimation of the input disturbance, the control offset vanishes. This shows that offset free control can be achieved with the disturbance estimator. It is also notable that the estimator is quite slow. This is due to the fact to the conservative choice of pole in the Lueneberger observer. It is necessary to keep the disturbance observer slower than the dynamics of the closed loop system. Otherwise the additional dynamics of the observer can cause instability. By keeping the observer slow, its dynamics can be considered constant compared to the systems dynamics and therefore they will not results in stability problems.

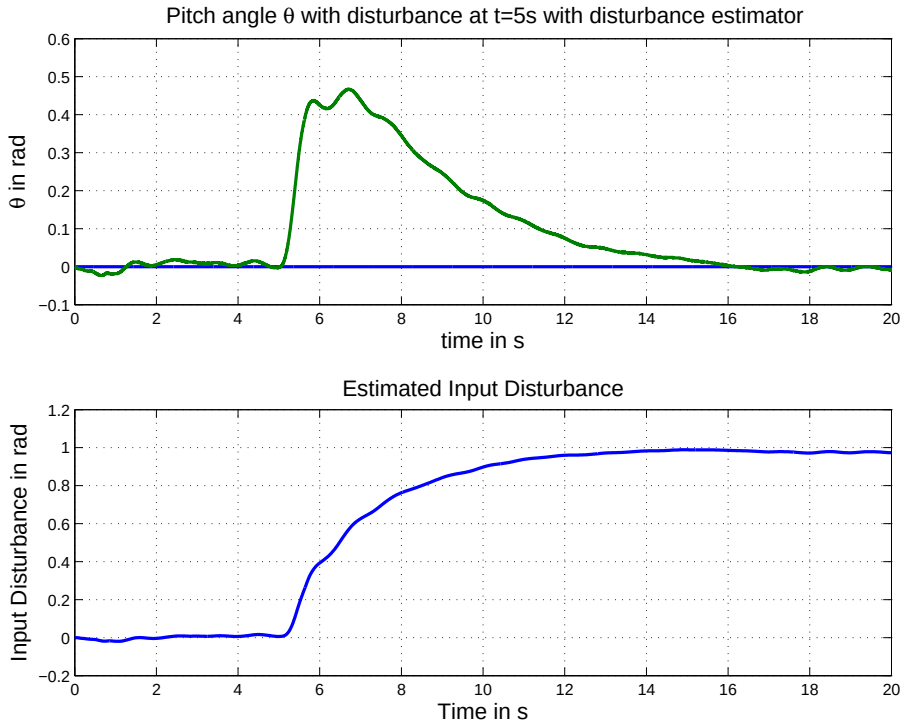

Figure 5.8: Pitch Angle with Disturbance at t=5s without Estimator.

5.1.2.2 Elevation Axis

There are also significant disturbances on the elevation axis. Like on the pitch axis, imperfections in the derived relation between impressed voltage and thrust cause an input disturbance. Moreover, the function for gravitational momentum has small errors, too.

In contrast to the pitch axis, the MPC actually provides offset free control in the presence of input disturbances without an input disturbance estimator. To show that, let's consider the following scenario: The MPC is to control the elevation angle to zero. The initial angle is $\phi_0 = -0.71rad$ (which means that the helicopter is laying on the ground). At $t_{dist} = 15s$ an input disturbance of $\Delta u_{diff} = -1N$ is added to the system. As you can see in Figure 5.9, the input disturbance is causing an initial offset at $t_{dist} = 15s$. In contrast to the pitch controller though, the MPC on the elevation axis is capable of compensating the control offset.

When the input disturbance estimator is added, the controller has almost the same performance (see Figure 5.10)

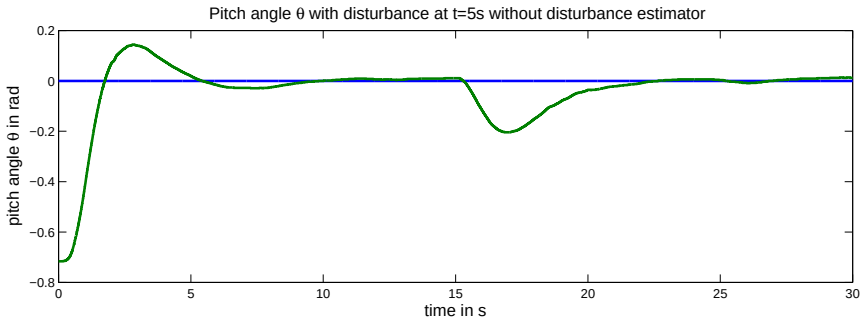

Figure 5.9: Elevation angle with Disturbance at t=15s without Estimator.

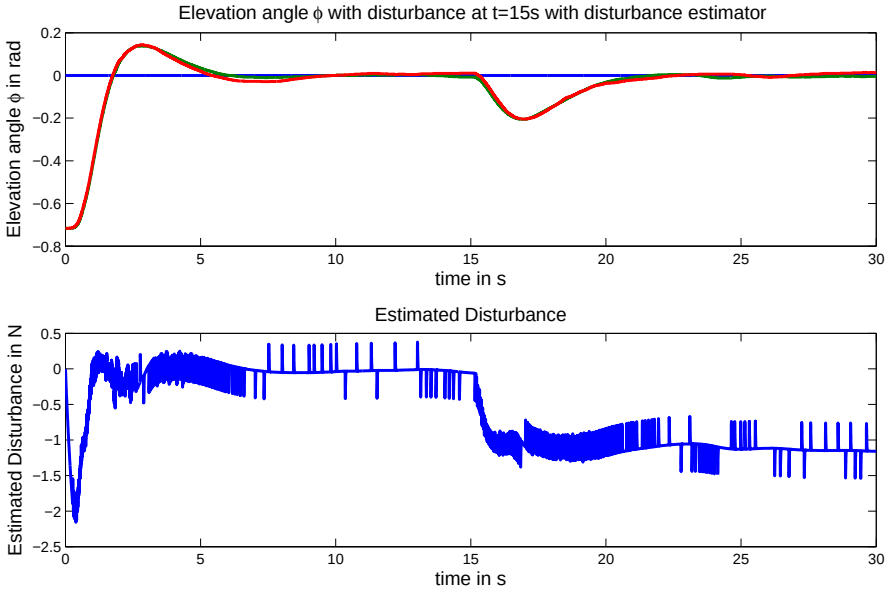

Figure 5.10: Pitch Angle with Disturbance at t=5s without Estimator.

5.2 Benchmark Scenario

The following flight path serves as the benchmark scenario: The helicopter starts on the ground ($\phi_0 = -0.7164rad$). From there, it moves up to a horizontal position ($\phi_1 = 0.0rad$). Then it turns $\frac{5}{2}\pi rad$ one and a quarter rotation in positive direction around the travel axis ($\psi_1 = \frac{5}{2}\pi rad$). There it descents to $\phi_2 = -0.30rad$ and ascents back $\phi_3 = 0.0rad$. From this position the helicopter turns back one rotation around the travel axis. There it descents back to $\phi_4 = -0.30rad$ and ascents again $\phi_5 = 0.0rad$ afterwards. From there the helicopter then turns back a quarter rotation around the travel axis to hover over the starting position. Finally, it descents to the starting position from there. These way points are also listed in Table 5.1 and depicted in Figure 5.11.

Table 5.1: Way Points Benchmark Scenario (ϕ, ψ in rad, Time in s).

Segment i	Time slot in s	ψ_i	ψ_{i+1}	ϕ_i	ϕ_{i+1}
1	$0-5$	0.0	0.0	-0.48	0.0
2	5 - 23	0.0	$\frac{5}{2}\pi$	0.0	0.0
3	23-24.5	$\frac{5}{2}\pi$	$\frac{5}{2}\pi$	0.0	0.0
4	$24.5-29.5$	$\frac{5}{2}\pi$	$\frac{5}{2}\pi$	0.0	-0.30
5	$29.5-31$	$\frac{5}{2}\pi$	$\frac{5}{2}\pi$	-0.30	-0.31
6	$31-36$	$\frac{5}{2}\pi$	$\frac{5}{2}\pi$	-0.31	0.0
7	$36-53$	$\frac{5}{2}\pi$	$\frac{1}{2}\pi$	0.0	0.0
8	$53-54.5$	$\frac{1}{2}\pi$	$\frac{1}{2}\pi$	0.0	0.0
9	$54.5-59.5$	$\frac{1}{2}\pi$	$\frac{1}{2}\pi$	0.0	-0.30
10	$59.5-61$	$\frac{1}{2}\pi$	$\frac{1}{2}\pi$	-0.30	-0.31
11	$61-66$	$\frac{1}{2}\pi$	$\frac{1}{2}\pi$	-0.31	0.0
12	$66-74.5$	$\frac{1}{2}\pi$	0.0	0.0	0.0
13	$74.5-74.9$	0.0	0.0	0.0	-0.48

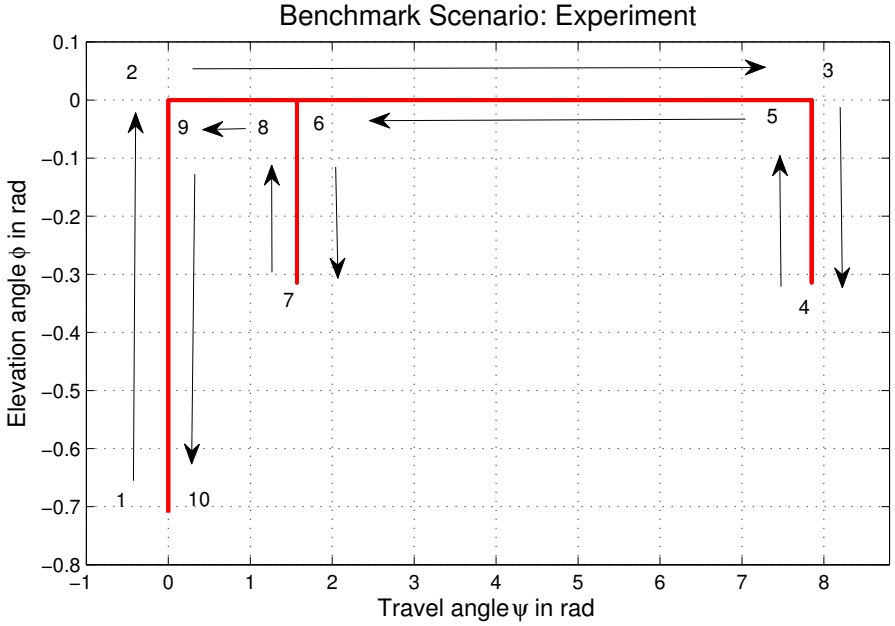

Figure 5.11: Benchmark Scenario.

5.2.1 Trajectory Planning

An important task in the benchmark scenario is to provide the controller with the reference target. In general, sufficiently smooth trajectories are necessary to achieve a good controller performance. In the case of MPC, in theory, this is not necessary, because the MPC is optimising the input signal to achieve an optimal trajectory. So the trajectory planning is included in the algorithm. In practise though, this is not necessarily true, because the prediction horizon is limited.

For the benchmark scenario, second degree polynomials where implemented between proceeding set points using the function smoothtrajectory (see Appendix A.1.2). The resulting trajectories are shown in Figure 5.12.

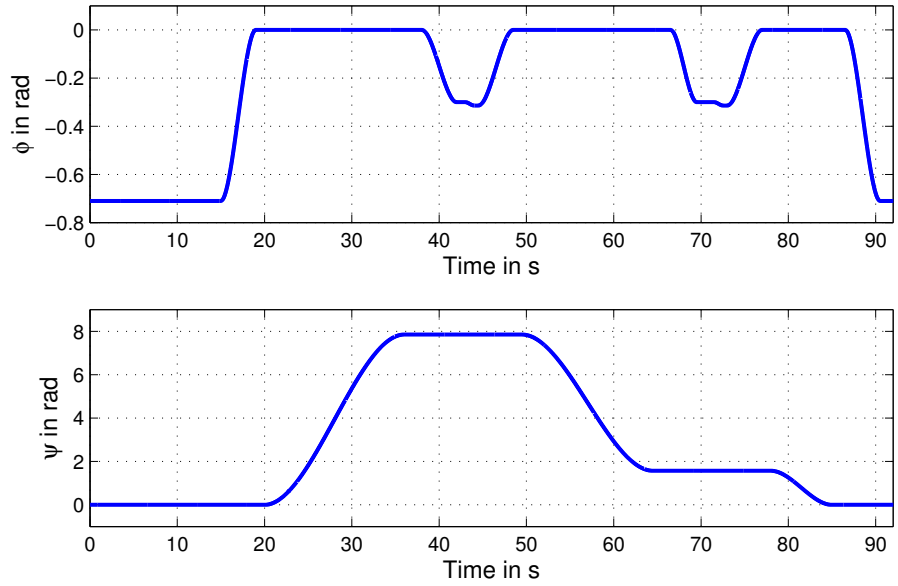

Figure 5.12: Benchmark Scenario.

5.2.2 Performance

The benchmark scenario has some challenging aspects for the controller. First of all, the controller needs to be sufficiently fast to follow the trajectory within the time limits. But higher speeds also make it more difficult for the controller to be accurate in terms of deviation from the reference trajectory. Accuracy is especially challenging on the elevation axis. This is due to the fact that the travel controller is using the pitch angle to control the travel angle. Changing the travel angle is changing the thrust on the elevation axis as well, which is acting as an input disturbance in the elevation axis. Here this problem was dealt with the input disturbance estimator.

The experimental results of the benchmark scenario without the use of an input disturbance estimator are shown in Figure 5.13. The first graph shows the experimental results in the θ-ϕ-plane, the red line represents the reference trajectory and the blue line the experimental results. It can be seen that there are some deviations in the elevation angle, especially close to the points where the helicopter is diving down (at $\psi = 0rad, \frac{1}{2}\pi, \frac{5}{2}\pi$). The reason is that the helicopter has to be slowed down on the travel axis in order to stop at these angles. This means that the pitch angle was changed which is causing an input disturbance on the elevation axis. This is difficult to deal with for the controller without

the use of the input disturbance estimator. The last three graphs show the angular behaviour on each axis over time for the benchmark scenario. The blue lines show the experimental results and the green line the reference trajectories. The pitch graph shows that the constraints on the pitch axis are actually hit on several occasions, mainly when the diving points are approached. During the first 15 seconds there is a considerable deviation from the reference on the pitch axis provided by the travel controller (blue line in the second graph in Figure 5.13 and Figure 5.14). This is caused by the fact that the controller is lying on the ground at this point, which makes it impossible to approach the reference value.

Figure 5.14 shows the experimental results for the benchmark scenario when the input disturbance estimator is activated. It is apparent on the first graph that the controller is now much more accurate on the elevation angle.

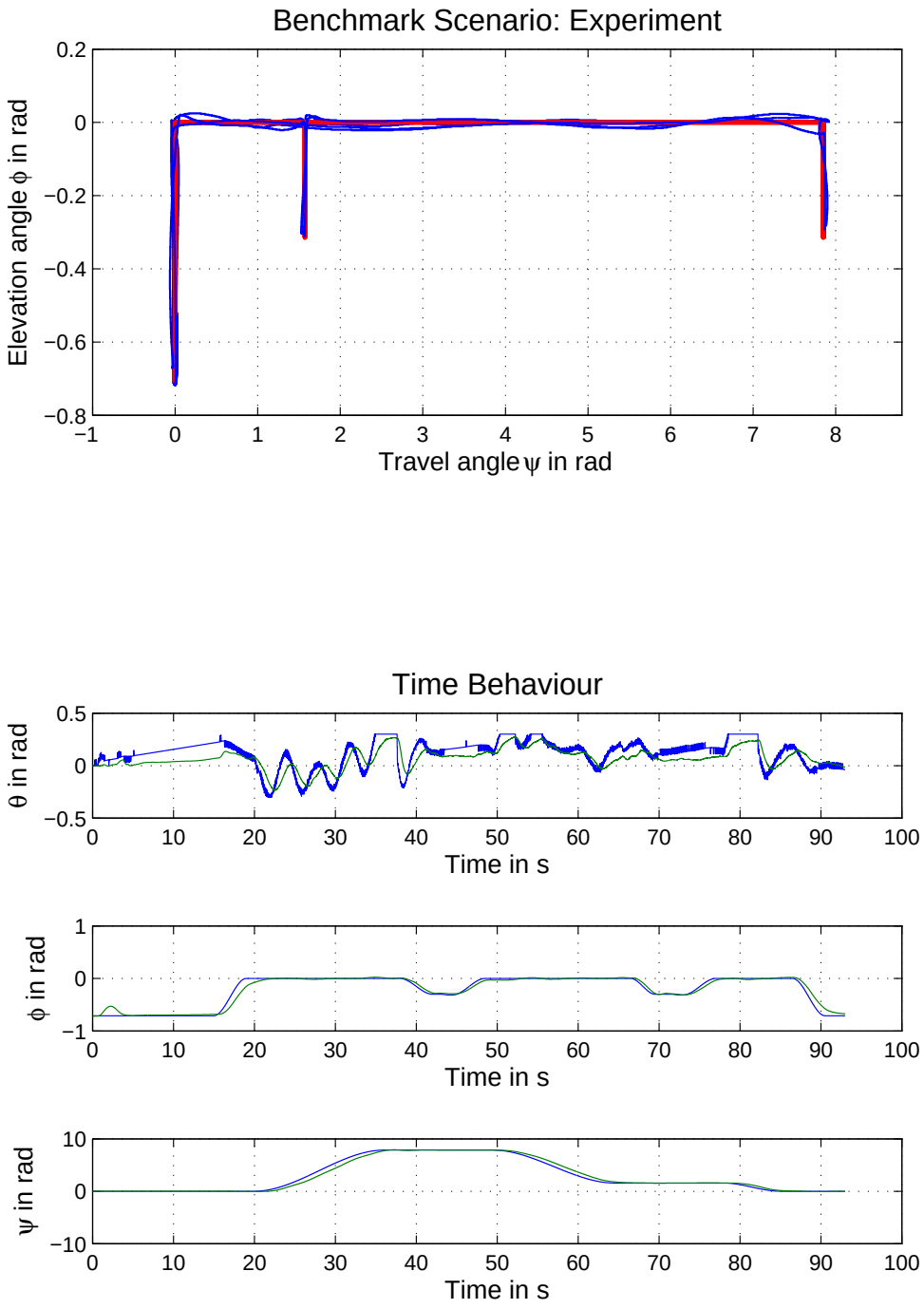

Figure 5.13: Benchmark Scenario with MPC, without Disturbance Estimation.

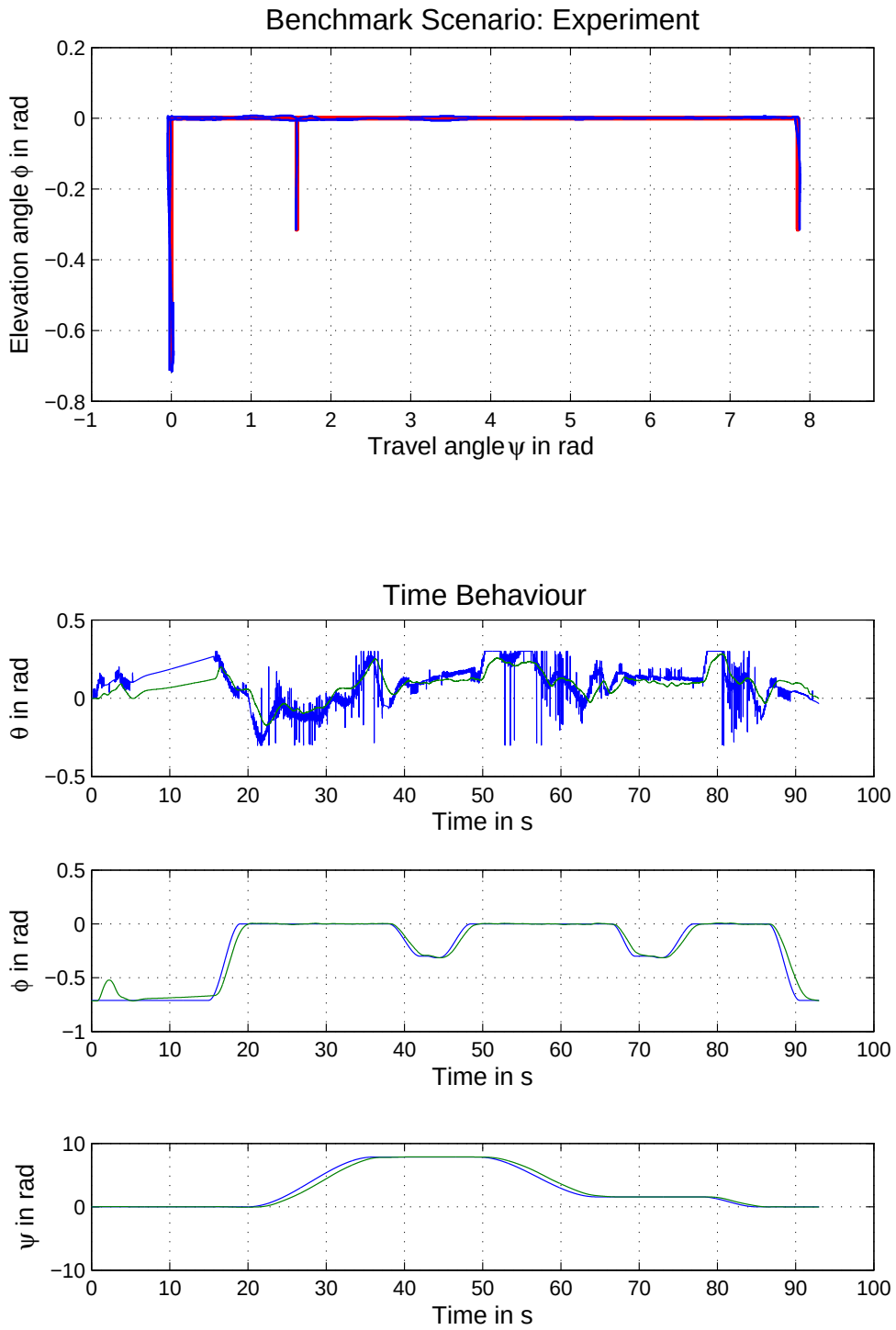

Figure 5.14: Benchmark Scenario with Disturbance Estimator.

Now the question is how the performance of the Model Predictive Controller compares to other kind of controllers implemented on the 3 DOF helicopter before. Cristian Wolf implemented a Linear Quadratic Regulator with flatness-based feed-forward in his thesis "Flachheitsbasierter Steuerungsentwurf am Quanser-Laborhelikopter (Flatness-based Feed-forward Design for the Quanser Laboratory Helicopter)" which he conducted at the Institute of System Theory and Control in Stuttgart in 2011. Figure 5.15 depicts his results of the benchmark scenario. Please note that he used degree instead of radiant. It is obvious that the deviation from the reference trajectory is significantly larger than for the Model Predictive Controller.

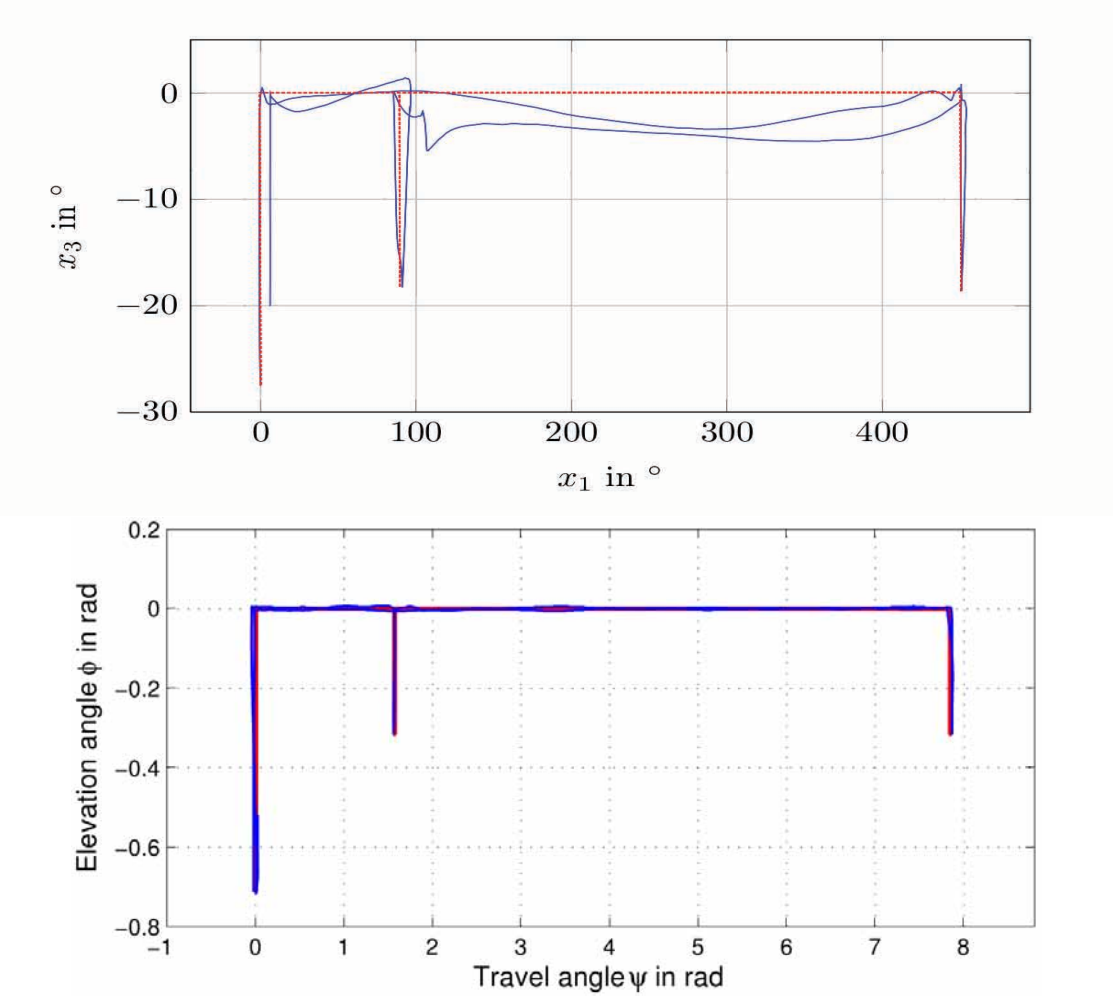

Figure 5.15: Benchmark Scenario Wolf (first graph) vs. MPC (second graph). Source: [Wol11], modified.

6 Summary and Conlusion

The topic of this thesis was the implementation and stability analysis of a Model Predictive Controller on a Quanser 3DOF helicopter. The first step was to identify a mechanical model using the principle of conservation of inertia. In that process, a relation between the impressed voltage on the motors and the rotor's thrust was identified. It turned out that the ground effect had an significant impact on the thrust. As the joint of the elevation axis of the helicopter is not in the center of gravity, the resulting gravitational momentum was identified using a weighting scale. A lower bound for the centrifugal force was identified as well. Additionally, in experiments the friction around the travel axis and the wind resistance dependent on the pitch angle were investigated. This resulted in a nonlinear state space model, which was verified using step experiments. In the next step, the nonlinear state space model was linearised and input as well as state constraints were defined. Then the model was cascaded into three individual models for each axis to be controlled by three individual Model Predictive Controllers. In the cascation structure there are two parallel loops: the elevation controller on the one hand and a loop consisting of the travel controller as an outer loop and the pitch controller as an inner loop on the other hand. Additionally, important nonlinear effects such as input nonlinearities and the centrifugal force were added as disturbances to the model. Thereafter, the Model Predictive Controllers were designed for each axis. The sampling intervals were chosen according to the cut-off frequencies of each axis and state and input tuned to achieve a good performance. For the pitch and elevation axis an input disturbance estimator was designed to handle zero-offsets caused by input disturbances such as disturbances in the motor's thrust. To ensure nominal stability, MPC stability theory was applied. Three different methods were investigated: the MPC with an Unconstrained Cost, the Dual Mode Model Predictive Controller and the Terminal State Method. For the MPC with Unconstrained Cost, the terminal weight was chosen as the solution of the Riccati Equation. In consequence, the MPC became equivalent to a Linear Quadratic Regulator as soon as the constraints were not active. Control and Prediction Horizon needed to have the same length for that though. This resulted in impractically small feasible sets. For the Dual Mode MPC, the idea was to switch the MPC to another controller as soon as the constraints are not active. Unfortunately, the structure of the MPC in this case was too complex to calculate this set with inactive constraints. Thus this method could not be applied. For the Terminal State Method, the terminal set was assumed to be only the origin. Because the prediction horizon could be chosen much longer than the control

horizon, rather large backwards reachable sets could be accomplished. Unfortunately, those sets were highly non-convex with very small convex subsets. Consequently, the feasible sets were very small as well. In the last part of the thesis, the performance of the implemented Model Predictive Controller was investigated. First, key properties such as time constant and settling time of the individual Model Predictive Controllers of each axis were evaluated. Additionally, the benefit of the input disturbance estimators on elevation and pitch axis were demonstrated. To show the overall performance of the Model Predictive Controller, a benchmark scenario was implemented. This scenario was used in thesis work and laboratories at University of Stuttgart before. In comparison, the Model Predictive Controller showed an excellent performance.

The goal of this thesis was to implement a Model Predictive Controller on the Quanser 3DOF helicopter which has fast dynamics. The results show that it is indeed possible to implement a controller which shows excellent performance compared to other types of controllers. The generally commended inherent constraint handling property of Model Predictive Control is actually the main asset in achieving this excellent performance. Nevertheless, it was abundant that limited computational power is still the most important issue when implementing Model Predictive Controllers on fast systems. Several measures such as the cascation of the model and the limitation of the horizon's length had to be taken to reduce the computational demand. This shows that the implementation of Model Predictive Controllers on fast system is possible, but can be very challenging. The second goal was to apply MPC stability theory on the 3DOF helicopter. It became quite apparent that there is a significant gap between MPC stability theory on paper and in practise. Many of the assumptions made in the theory are actually very limiting and often very difficult to meet. There are two main aspects which turned out to be most problematic: MPC stability theory is based on the assumption that the origin or a terminal set can be reached within the MPC's prediction horizon. This might not be possible from the whole state space, but only from a limited feasible set. The size of this feasible set mainly depends on the length of the control and prediction horizon of the MPC. In the case of the 3DOF helicopter, those horizons were very limited due to the fast dynamics of the system. In consequence, the MPC stability theory resulted in impractically small feasible sets. The second practical problem was to find the terminal constraint sets. They are assumed to be known in MPC stability theory, but it is not clear how they can actually be calculated in practise. There was no algorithm available to calculate a terminal set directly. Instead, the terminal constraint sets were calculate for the unconstrained MPC using the Multi-Parametric Toolbox. This did not always work properly. Another problem was that the resulting terminal constraint sets were often very small or could not be calculated at all. Those difficulties prevented the usage of a Dual Mode Model Predictive Controller in this application. It is clear that both the implementation of the MPC as well as the stability theory would benefit from more computational power or a more efficient implementation of the controller. The implementation of the 3DOF helicopter

in a HIL system is certainly not optimal computationally. Nevertheless, the 3DOF helicopter shows that computational limitations still remain the most important challenge when implementing a Model Predictive Controller on fast systems. It was still possible though to implement a Model Predictive Controller on the Quanser 3 DOF helicopter and achieve an excellent performance. This shows that with even more computational power and better optimisation algorithms, Model Predictive Control becomes an excellent alternative for many application outside the process industry.

A Appendix

A.1 Implementation

A.1.1 Backwards Reachable Set

In order to calculate the backwards reachable set from a given set x, the following function has been implemented in Matlab:

```
function [y] = back_reach_set(x,Ainv,B,Nu,Umax,Umin)
y=x;
[a, xlength] = size(x);
intu = (Umax-Umin)/Nu
for i = 1:1:xlength
    for j=0:1:Nu
            y = [y, Ainv*y(:,i)-Ainv*B*(intu*j+Umin)];
    end
end
```

where x is the initial set, Ainv the inverted system matrix, B the input matrix, Nu the resolution of the discretisation of possible input signals, Umax the maximal input and Umin the minimal input.

The function delivers a new set y which contains reachable points within one step.

A.1.2 Trajectory Planning

To get sufficiently smooth trajectory, the function smoothtrajectory was implemented to generate a second degree polynomial between two setpoints:

```
%% Generates a second degree polynomial between two setpoint and
%  returns a set of points of the generated polynomial
%           t is a vector containing start and end time
%           f contains the setpoints
%           f1 contains the desired gradient on the setpoints
%           tp is a vector describing the point of time for the
```

```
%           returned set of points describing the polynomial

function [points] = smoothtrajectory(t,f,f1,tp)

lgs = [     t(1)^3,t(1)^2,t(1), 1;
            3*t(1)^2,2*t(1),1, 0;
            t(2)^3,t(2)^2,t(2), 1;
            3*t(2)^2,2*t(2),1, 0    ];
b =   [     f(1);
            f1(1);
            f(2);
            f1(2)    ];

coef = inv(lgs)*b;
points = coef(1)*tp.^3 + coef(2)*tp.^2 +coef(3)*tp.^1 +coef(4)*tp.^0;
```

A.2 Multi-Parametric Toolbox (MPT)

The Multi-Parametric Toolbox (MPT) is a Matlab toolbox developed by researchers at the Automatic Control Laboratories at ETH Zürich [mpt]. It offers a wide range of algorithms for the design, analysis and implementation of optimal controllers for constrained systems.

The toolbox also includes an algorithm to calculate the invariant set of an MPC for the specified constrained system:

```
invCtrl = mpt_invariantSet(ctrl, Options)

%MPT_INVARIANTSET Computes (robust) positive invariant subset of an %
   explicit controller
```

Unfortunately, this function only works for explicit Model Predictive Controllers. In order to use the toolbox for regular MPC used in this thesis, a subfunction of mpt_infset was used:

```
 [Oinf,tstar,fd,isemptypoly] = mpt_infset(A,X,tmax,Pnoise,Options)

% Calculates the maximal positively invariant set for an autonomous
% discrete-time  LTI system  Takes into account polytopic and additive
% system uncertainty, if defined  in "sysStruct" / "Pnoise",
% respectively.
```

This function is able to calculate the invariant set of an autonomous system. In order to use it for a Model Predictive Controller, a state space representation of the MPC has to be derived. In general, this is only possible for an unconstrained MPC. With the state space representation of the MPC the resulting autonomous system can then be calculated.

Bibliography

[con] http://upload.wikimedia.org/wikipedia/commons/1/11/MPC_scheme_basic.svg,29.08.2013

[Doh11] Dohnal, Adam: Modellierung, Identifikation und Regelung eines Hubschrauber-modells mit drei Freiheitsgraden. In: *Institut fuer Systemtheorie und Regelungstechnik* (2011)

[Eit11] Eitel, Andreas: Entwicklung eines Simulators fuer einen Laborhelikopter. (2011)

[Ekh13] Ekhard, Bo: Lecture Notes Model Predictive Control. (2013)

[exp] http://www.quanser.com/products/3dof_helicopter,13.09.2013

[Löf01] Löfberg, Johan: Linear Model Predictive Control Stability and Robustness. (2001)

[Mac00] Maciejowski, Jan: *Predictive Control with Constraints*. Prentice Hall, 2000

[mpt] http://control.ee.ethz.ch/~mpt/2/,24.09.2013

[Sco00] Scokaert, D.Q. Mayne J.B. Rawlings C.V.Rao P.: Constrained model predictive control: Stability and optimality. In: *Automatica* 36 (2000), S. 789–814

[Ste03] Stevens, Frank Brian L. Brian Lewis: *Aircraft Control and Simulation*. Jon Wiley sons, 2003

[Wol11] Wolf, Christian: Flachheitsbasierter Steuerungsentwurf am Quanser-Laborhelikopter. (2011)